Centro de Investigaciones en Óptica AC

Maestria en Optomecatrónica

TESIS

SELECCIÓN Y MANIPULACIÓN DE OBJETOS A PARTIR DE SU FORMA GEOMÉTRICA Y COLOR POR MEDIO DE UN BRAZO ROBÓTICO

PARA OBTENER EL GRADO DE:

MAESTRO EN OPTOMECATRÓNICA

PRESENTA

Ing. César Paúl Carrillo Delgado

Asesor: Dr. J Ascención Guerrero Viramontes

León, GTO, Diciembre de 2010

Agradecimientos

Agradezco al Consejo Nacional de Ciencia y Tecnología ***CONCACYT*** por brindarme una beca de 2 años para que fueran posibles los estudios de maestría.

Agradezco también al Centro de Investigaciones en Optica A.C (**CIO**) por darme su apoyo en muchas cosas desde el inicio del curso propedéutico hasta el final de la Maestría en Optomecatronica, la cual no hubiera sido posible sin toda la atención que se dio a mi proyecto de tesis.

Al **Dr. Cuevas** (Director de Formación Académica) por haber tenido la visión de equipar muy bien el laboratorio en el cual trabajé con el Robot KUKA KR16, el cual es parte importante de mi trabajo de tesis.

Al **Dr. Ascencion**, el cual además de asesor de tesis fue un guía en un medio en el cual nunca había estado, y con esto poder llegar a realizar un excelente trabajo final.

También quisiera agradecer a todos los **maestros** que me dieron clases durante la maestría y me ayudaron a mejorar en gran medida con respecto a cuando llegue al CIO en el aspecto personal y académico.

Dedicatoria

A mis padres y tíos, *por su apoyo para que todo fuera posible.*

A mi gran familia, *la cual hacia todos los esfuerzos posibles para que yo pudiera continuar estudiando.*

A mis amigos y compañeros del CIO, *con los que pasamos fuertes batallas tanto en los cubos como en el laboratorio y los que de alguna manera se vieron involucrados en mi trabajo de tesis.*

INDICE

CAPITULO 3. RECONOCIMIENTO DE FORMA

CAPITULO 4. SISTEMA ROBOTICO KUKA KR-16

CAPITULO 5. CONCLUSIONES GENERALES

JUSTIFICACIÓN DE TESIS

Cuando los trabajos para el humano se hacen complicados y tediosos además de repetitivos por tiempos prolongados incluyendo riesgos considerables para su integridad, se pueden generar alternativas para automatizar dichas tareas , entre las cuales se puede mencionar algunas como la sustitución de la visión humana por la de un robot el cual puede tomar decisiones dependiendo de una variable de entrada como podría ser una imagen, se puede mencionar también la sustitución de la fuerza de alguna persona por la de un brazo robótico el cual sea diseñado para sujetar objetos de gran tamaño y peso, evitando con esto que pueda sufrir algún tipo de lesión debido a la frecuencia y tiempo con la que se realiza dicha tarea; cabe señalar que un robot puede realizar casi las mismas tareas que las de un humano común, la gran diferencia radica en la gran exactitud y repetitividad que dicho dispositivo se puede mantener realizando la misma tarea por muchas más horas que las que un humano podría soportar sin cansancio mental y físico que lo podrían llevar a aumentar el riesgo de ser víctima de su propio cansancio , llevándolo con esto a realizar su trabajo de una mala manera además de estar cada vez más cerca de sufrir alguna lesión que lo pudiera marginar de sus obligaciones diarias. También es preciso mencionar que la inversión inicial para un brazo robótico es cara, pero es un tipo de inversión que a mediano y largo plazo será redituable para optimizar la producción de grandes cantidades de productos en el menor tiempo posible.

Generación de beneficios

Los beneficios más importantes de esta automatización están dados por la capacidades de exactitud, detección, manipulación, fuerza y gran rapidez, que se conjuntan para lograr un objetivo en común que es la selección de objetos a partir de su forma, aplicable en las industrias de muchos rubros como lo son el calzado, automotriz, juguetera, farmacéutica, así como poder ser utilizado por personas que no estén en algún área antes mencionada, sin la necesidad de tener conocimientos técnicos previos de electrónica, control, robótica o programación debido a su fácil manejo, implementación y gran aplicabilidad.

Se menciono en el párrafo anterior algunos de los tantos beneficios técnicos que este proyecto puede brindar, pero sería bueno especificar los beneficios en cuanto a que esta aplicación se le puede dar un uso para el cual no fue diseñado, si no que se puede implementar en muchas áreas donde se necesite detectar formas de objetos en base a un patrón de comparación establecido que puede ser definido por la persona , lo que lo hace aplicable en una amplia gama de procesos dado que puede detectar formas perfectas y también aquellas que se aproximen a dicho patrón. Pudiendo con esto detectar errores en objetos que no tengan la forma deseada, para que posteriormente los objetos que cumplan con la forma deseada sean aceptados y empacados, así como los objetos malformados puedan ser descartados por el manipulador par ser desechados y mantener un control de calidad dentro de un rango determinado por los usuarios.

Desde el punto de vista práctico

Desde el punto de vista práctico este proyecto logra optimizar procesos tomando en cuenta la rapidez y exactitud con la que realiza la selección de objetos lográndolos colocar en la coordenada programada en un tiempo menor al que lo haría una persona normal, logrando con esto una mayor productividad a corto plazo.

Titulo de la tesis: SELECCIÓN Y MANIPULACIÓN DE OBJETOS A PARTIR DE SU FORMA GEOMETRICA POR MEDIO DE UN BRAZO ROBOTICO.

Defensor: Ing. César Paúl Carrillo Delgado
Asesor: Dr. J. Ascención Guerrero Viramontes
Posgrado: Maestría en Optomecatrónica

RESUMEN

Este trabajo está enfocado en la selección de objetos tomando en cuenta su forma y color para, posteriormente, ser manipulados por un brazo robótico, con esto se puede lograr la automatización en procesos de empaque, control de calidad por detección de forma, así como la sujeción y traslado de objetos pesados, entre otros, logrando con esto evitar el contacto directo del ser humano en esta etapa del proceso.

Este desarrollo tecnológico está compuesto por tres partes fundamentales: El sensor de detección de forma, el modulo de comunicación y el robot manipulador, de estos se desprenden mas módulos pequeños los cuales se fusionan con los anteriores para conformar la totalidad de esta automatización; las pruebas para la funcionalidad de las partes mencionadas fueron probadas primeramente por separado, ya que todas ellas fueron diseñadas especialmente para este proyecto, aunque pueden ser aplicadas para otros proyectos dado construcción e interacción con el robot.

La primera parte fundamental de esta automatización es el sensor de detección de forma realizado en el laboratorio, el cual consta de una cámara y un algoritmo de procesamiento de imágenes, el cual analiza ciertas variables antes de llegar al resultado final, que sería la detección sin error de la forma de la figura.

La segunda parte importante es el modulo de comunicación, el cual tiene como primer objetivo recibir los datos del sensor que son enviados por el puerto USB y codificados mediante la librería de comunicación de USB CDC para ser enviados a la unidad de control del brazo robótico el cual recibe los datos por medio de un modulo de

entradas y salidas digitales. El segundo objetivo de esta parte es el de acoplar y aislar el circuito de control mediante relevadores del modulo de entrada antes mencionado.

La última parte principal es el brazo robótico y su programación, encargados de manipular las piezas seleccionadas mediante un gripper neumático, además de ubicar la cámara por encima de la figura que va a ser detectada.

CAPITULO 1
Introducción

1.1 Importancia de la robótica

El desarrollo de la ciencia y la tecnología ha marcado pasos gigantescos en el progreso de la sociedad y como ejemplo se puede mencionar los robots para el manejo de herramientas (medicina, investigación, hogar), una aplicación de ellos, se enfoca en las cirugías de los ojos, extirpar tumores, así como en el ligamento de cartílagos, por mencionar algunas. Otras áreas que también se pueden citar son otras como la agricultura, la mecánica o la industria. También cabe mencionar que el avance de la robótica ha favorecido y favorece al incremento de la productividad de alimentos, ropa, calzado, aparatos electrónicos, en la elaboración de perfumes, etc.

El ser humano, por la necesidad de facilitar y agilizar los procesos de producción y obtener mayores ganancias económicas construyó El ROBOT, el cual se define de la siguiente forma. Es una unidad programable diseñada para mover materiales, piezas, herramientas o dispositivos especiales, por medio de movimientos variables programados para la ejecución de diferentes trabajos"[1].

Además, a diferencia del ser humano éste no se cansa y su funcionamiento es dirigido por un cerebro (una computadora) programado para que realice las tareas.

La robótica tiene como intención final complementar o sustituir las funciones del ser humano, alcanzando en algunos sectores, aplicaciones masivas. En el contexto industrial, donde se utilizan con notable éxito desde hace varias décadas, sus beneficios empresariales y sociales se pueden resumir en 4:

1. **Productividad:** Significa aumento de la producción así como reducción de costos de mano de obra, materiales, energéticos y de almacenamiento.

2. **Flexibilidad:** Permite la fabricación de productos del mismo tipo por mucho tiempo sin la necesidad de parar, por cansancio o cambio de turno la producción y, por consecuencia, se reducen tiempos muertos y aumenta la eficiencia.

3. **Calidad:** Debido al alto nivel de repetitividad en las tareas realizadas por los robots, se asegura una calidad uniforme del producto final.

4. **Seguridad:** Debido a que los procesos de fabricación se llevan a cabo con un número mínimo de personas, disminuyen las posibilidades de accidentes laborales, además de reemplazar a los operadores de tares tediosas y cansadas.

Por otro lado, cabe señalar que la robótica ofrece grandes beneficios sociales: resolviendo problemas cotidianos que abarcan desde tareas simples pero tediosas en el hogar hasta la industria de ensamble de piezas a gran escala, lo cual se hace presente en todos los sectores y niveles de la población, mejorando la calidad de vida de los ciudadanos al reducir las horas de trabajo y riesgos laborales. También aporta beneficios económicos aumentando la competitividad de las empresas, favoreciendo su crecimiento así como creando nuevos negocios y profesiones.

Los sectores a los que está orientada actualmente la robótica son muy extensos empezando en las industria manufacturera (automotriz, manejo de herramientas, empacado de productos, control de calidad, etc.) hasta la exploración de ambientes hostiles, tales como entornos submarinos, el espacio, así como todos los lugares inaccesibles para el ser humano debido al alto grado de peligrosidad que esto puede representar. No obstante, la robotización no solamente tiene fines industriales sino también múltiples aplicaciones sociales, tales como, asistencia personal, medicina, limpieza, inspección, y mantenimiento. Sería oportuno señalar que en la actualidad la robótica se divide en 2 grandes áreas, la robótica industrial y la de servicio; aunque la robótica industrial en México es más conocida y, está un poco mejor establecida, que la de servicios, ambas representan grandes posibilidades de investigación y desarrollo que dan lugar a la robótica avanzada.

En relación a la **robótica industrial** es bueno resaltar que la mayoría de los robots están instalados en la industria manufacturera, en concreto, alrededor del 50% están en la industria del automóvil. Las aplicaciones más demandadas son las de soldadura y manipulación que en los países más industrializados son del orden del 25-50% y 30-60%, respectivamente dependiendo del país. Es interesante destacar que, mientras

aproximadamente el 40% de robots industriales se encuentran en Japón, los lideres en la fabricación son empresas europeas (ABB, KUKA, COMAU, STAUBLI) que dominan casi la totalidad del mercado.

Por otro lado, la **robótica de servicio** es un campo emergente pero con gran potencial de crecimiento. Sus aplicaciones se dividen en servicios personales (asistencia a personas mayores, discapacitados y niños, limpieza y seguridad domestica) y servicios profesionales (Limpieza, vigilancia, mantenimiento, inspección, medicina, construcción y agricultura). La mayoría de los sectores y aplicaciones citadas cuentan con un muy bajo nivel de automatización ocupando un gran número de trabajadores en actividades tediosas y peligrosas.

1.2Robot Manipulador

Un **robot industrial** es un manipulador multifuncional reprogramable que consiste en una secuencia de cuerpos rígidos llamados elementos que están conectados mediante articulaciones prismáticas o de revolución. Cada par, articulación-elemento constituye en grado de libertad. Estos manipuladores son capaces de mover materias, piezas, herramientas, o dispositivos especiales, según trayectorias variables, programadas para realizar tareas diversas.

Las articulaciones y elementos se enumeran hacia afuera desde la base; así que la articulación 1 se le puede denominar al punto de conexión 1 y la base de soporte (elemento 1). Cada elemento se conecta a lo mucho a otros 2 para evitar formar lazos cerrados.

En la figura 1.1 se presenta el brazo manipulador industrial utilizado en este proyecto (KUKA-KR16) de 6 grados de libertad tomando en cuenta el actuador final (**Gripper**) encargado de realizar la sujeción, el cual no se muestra en esta figura, este actuador se encuentra en el eje A6 lo que lo hace entrar en la categoría de un brazo manipulador.

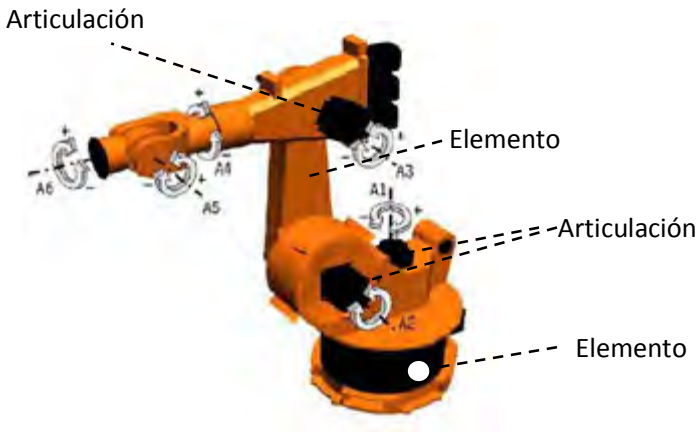


Figura 1.1 Brazo manipulador con 6 grados de libertad

1.2.1 Ventajas y Desventajas de los Robots

Ventajas

- Mayor precisión sin cansancio.
- Realización de tareas peligrosas.
- Reducción de costos a largo plazo
- Puede acceder a lugares que para el ser humano son de difícil acceso.
- Fuerza para el levantamiento de objetos.
- Mayor rapidez sin perder precisión que el ser humano pierde al cansarse

Desventajas

- Peligrosos según la aplicación.
- Desplazamiento de la mano de obra humana.
- Se necesita una gran inversión inicial.
- Necesitan mantenimiento periódico.

1.3 Componentes de los Robots

Es importante mencionar, que aunque el propósito esencial de un robot industrial es el de reemplazar al hombre en la realización de ciertas tareas, la configuración de su estructura mecánica no pretende imitar la humana. A pesar que se hable de brazo, muñeca o mano, no debemos de caer en el error de limitar al robot con características humanas (“la

réplica más que estructural es funcional"). Todos los componentes se mencionan a continuación y se pueden observar en la figura 1.2.

- *Estructura Mecánica.*- Un robot está formado por eslabones que van unidos entre sí por articulaciones, que a su vez se asemejan a un brazo humano, motivo por el cual se usan palabras como brazo, codo, y muñeca. De esta forma se puede dar el movimiento entre dos eslabones consecutivos.
- *Transmisiones*.- Son los que transmiten el movimiento del actuador hasta la articulación.
- *Actuadores.*- Generan el movimiento del robot, estos pueden ser: neumáticos, hidráulicos o eléctricos.
- *Sistema Sensorial.*- Es el encargado de darle información al robot de su propio estado (sensores internos) y el de su entorno (sensores externos).
- *Sistema de Control.*- Es el encargado de regular el comportamiento del robot para obtener los resultados deseados.
- *Actuadores Finales.*- Son los que interactúan directamente con el entorno, generalmente son diseñados específicamente para cada tipo de trabajo.

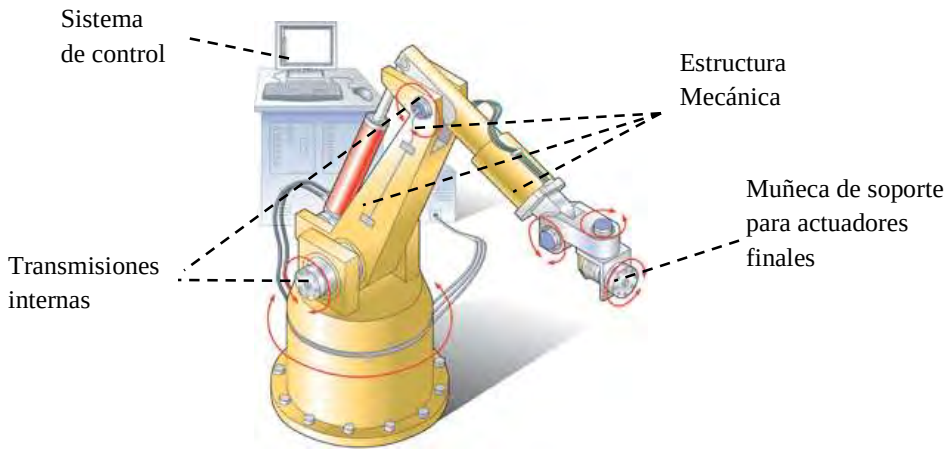


Figura 1.2 Brazo manipulador de herramienta donde señalan sus componentes sin el sistema sensorial que se mencionara más adelante.

1.4 Clasificación del robot Industrial

La maquinaria estática para la automatización de procesos dio paso al robot, con el desarrollo de controladores rápidos, basados en el microprocesador, así como un empleo de servos en lazo cerrado, que permiten establecer con exactitud la posición real de los elementos del robot y establecer el error con la posición deseada. Esta evolución ha dado origen a una serie de tipos de robots, que se citan a continuación:

1.4.1 Manipuladores

Son sistemas mecánicos multifuncionales, con un sencillo sistema de control, que permite gobernar el movimiento de sus elementos, de los siguientes modos:

- Manual: Cuando el operario controla directamente la tarea del manipulador.
- De secuencia fija: cuando se repite, de forma invariable, el proceso de trabajo preparado previamente.
- De secuencia variable: Se pueden alterar algunas características de los ciclos de trabajo.

Existen muchas operaciones básicas que pueden ser realizadas óptimamente mediante manipuladores, por lo que se debe considerar seriamente el empleo de estos dispositivos, cuando las funciones de trabajo sean sencillas y repetitivas. Un ejemplo se muestra en la figura 1.3.

igura 1.3 Brazo manipulador de herramienta

1.4.2 Robots de repetición o aprendizaje

Son manipuladores que se limitan a repetir una secuencia de movimientos, previamente ejecutada por un operador humano, haciendo uso de un controlador manual o un dispositivo auxiliar. En este tipo de robots, el operario en la fase de enseñanza, se vale de un panel de programación con diversos pulsadores o teclas, o bien, de joysticks, o bien utiliza un maniquí, o puede desplazar directamente la mano del robot. Los robots de aprendizaje son los más conocidos, hoy día, en los ambientes industriales y el tipo de programación que incorporan, recibe el nombre de "gestual"[2] y [3].

1.4.3 Robots controlados por computadora

Son manipuladores o sistemas mecánicos multifuncionales, controlados por una computadora, que habitualmente suele ser una microcomputadora.

En este tipo de robots, el programador no necesita mover realmente el elemento de la maquina, cuando la prepara para realizar un trabajo. El control por computadora dispone de un lenguaje especifico (como en el caso de KUKA KR-16), compuesto por varias instrucciones adaptadas al robot, con las que se puede confeccionar un programa de aplicación utilizando solo la salidas de la computadora, no el brazo. A esta programación se le denomina textual y se crea sin la intervención del robot manipulador.

Las grandes ventajas que ofrecen este tipo de robots, hacen que se vayan imponiendo en el mercado rápidamente, lo que exige la preparación urgente de personal calificado, capaz de desarrollar programas similares a los de tipo informático.

1.4.4 Robots inteligentes

Son similares a los del grupo anterior, pero además, son capaces de relacionarse con el mundo (Figura 1.4) que les rodea a través de sensores y tomar decisiones en tiempo real (auto programable). De momento, son muy poco conocidos en el mercado y se encuentran en fase experimental, en la que se esfuerzan los grupos de investigación por potenciarles y hacerles más efectivos, al mismo tiempo que más accesibles en cuanto a precio.

La visión artificial así como la inteligencia artificial, son las ciencias que más están estudiando para su aplicación en los robots inteligentes.

propias

1.4.5 Microrobots

Con fines educacionales, de entretenimiento o investigación, existen numerosos robots de formación o micro-robots a un precio muy accesible, cuya estructura y funcionamiento son similares a los de aplicación industrial.

1.5 Programación del Robot

Un gran obstáculo en la utilización de los manipuladores como máquinas de uso general es la falta de comunicación eficaz y apropiada entre el usuario y el sistema robótico, de forma que éste pueda dirigir al manipulador para cumplir una tarea dada. Hay algunas formas de comunicarse con un robot, y los tres grandes enfoques para lograrlo son: el reconocimiento de palabras, enseñar y reproducir (explicito), incluyendo los lenguajes de programación de alto nivel.

La programación empleada, como se menciono, puede tener un carácter ***explícito***, en el que el operador es el responsable de las acciones de control y de las instrucciones adecuadas que se implementan, o también puede estar basada en la modelación en base a un mundo exterior, cuando se describe la tarea y el entorno, tomando el sistema sus propias decisiones.

La programación ***explícita*** es la utilizada en las aplicaciones industriales y consta de dos técnicas fundamentales:

A. Programación Gestual.
B. Programación Textual.

La ***programación gestual*** consiste en guiar el brazo del robot directamente a lo largo de la trayectoria que debe seguir. Los puntos del camino se graban en memoria y luego se repiten. Este tipo de programación, exige el empleo del manipulador en la fase de enseñanza, o sea, trabaja "on-line"(Figura 1.5).

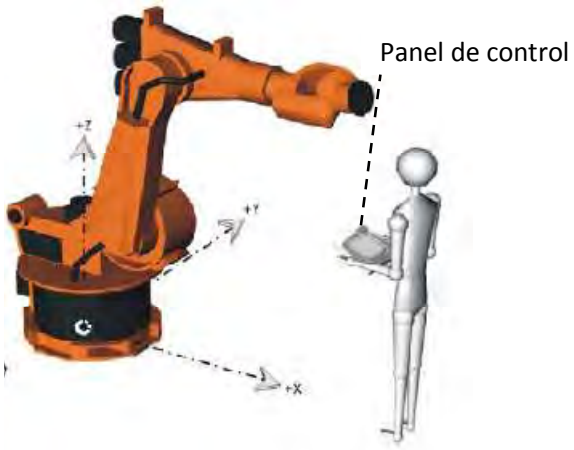


Figura 1.5 En la programación gestual el usuario recorre manualmente las trayectorias con un panel de control para después grabarlas y generar el programa

Para ser más concreto el 50% del programa del robot para esta automatización esta hecho por medio de un tipo de programación gestual, llamada programación mediante un dispositivo de enseñanza, que consiste en determinar las acciones y movimientos del brazo manipulador, a través de un elemento especial para este cometido. En este caso, las operaciones ordenadas se sincronizan para conformar el programa de trabajo.

El dispositivo de enseñanza suele estar constituido por botones, teclas, pulsadores, luces indicadoras, ejes giratorios o "joystick" que para el caso del KUKA KR16 es llamado KCP(KUKA Control Panel), el cual se puede observar en la figura 1.6.

Figura 1.6 Panel de control utilizado como dispositivo de enseñanza

Dependiendo del algoritmo de control que se utilice, el robot pasa por los puntos finales de la trayectoria enseñada. Hay que tener en cuenta que los dispositivos de enseñanza más actuales no sólo permiten controlar los movimientos de las articulaciones del manipulador, sino que pueden, también, generar funciones auxiliares, como:

- Selección de velocidad
- Selección de orientación de herramienta
- Generación de retardos
- Señalización del estado de los sensores
- Borrado y modificación de los puntos de trabajo
- Selección de sistemas de coordenadas.

En la ***programación textual***, las acciones que ha de realizar el brazo se especifican mediante las instrucciones de un lenguaje. En esta labor no participa la máquina (off-line). Las trayectorias del manipulador se calculan matemáticamente con gran precisión y se evita el posicionamiento por cálculo a simple vista.

Este tipo de programación es el otro 50% del algoritmo implementado y permite realizar operaciones más complejas y con mayor grado de precisión. Además, presenta la ventaja de que es posible establecer relaciones entre el robot y su entorno, por ejemplo mediante una cámara. Para ello basta con introducir en el programa los datos procedentes de los sensores de forma, que el robot actúe en constante comunicación los mismos, tal y como ocurre en los denominados robots inteligente.

A su vez, la programación textual puede ser de dos tipos, explícita y especificativa, la cuales se definen a continuación.

- La **programación textual explícita (Figura 1.7)** se corresponde con los llamados lenguajes estructurados. Consiste en programar de forma secuenciada y estructurada el conjunto de acciones que debe realizar el robot para llevar a cabo la tarea encomendada. En dichas instrucciones pueden introducirse también las características del medio.
- La **programación textual especificativa** está más relacionado con los lenguajes de programación orientados a objetos. En este caso, el programa gira en torno a los elementos manipulados por el robot y las acciones que ha de realizar con ellos, teniendo en cuenta el ámbito en el que se desarrollan dichas acciones.

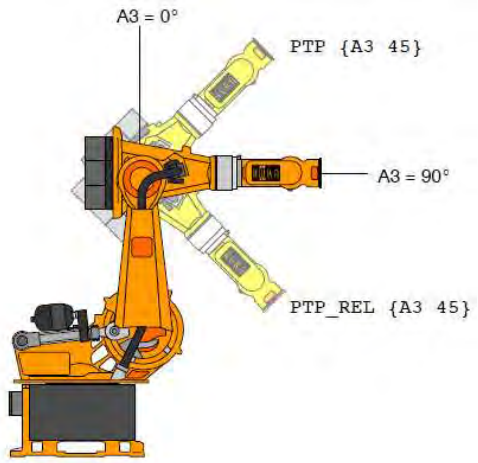
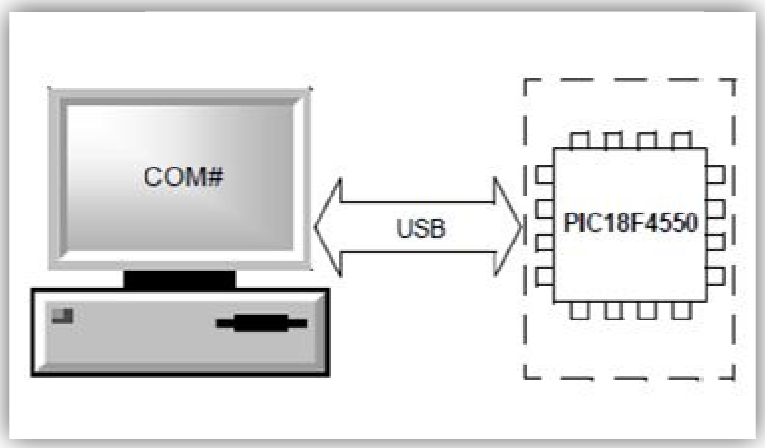


Figura 1.7 Movimientos propios de la programación mediante instrucciones (programación textual) utilizando ángulos de rotación

1.6 Comunicación puerto USB-ROBOT

Para que el robot pudiera establecer comunicación con el sensor encargado de detectar la forma se usa el puerto USB incluido en una computadora, pero cuando se le conecta a ella misma el modulo de comunicación (microcontrolador y relevadores para acoplar), que se explicara más adelante, este puerto es detectado (con la instalación previa de un driver especifico para el sistema operativo) como un puerto serie virtual, el cual a su

vez es controlado por el programa de reconocimiento de forma realizado en MATLAB al cual se le programa el algoritmo para enviar los datos como si se tratara de un puerto serial como se muestra en la figura 1.8.



del modulo de
puerto serial

1.6.1 USB

El **USB** (Bus de serie universal), como su nombre lo dice, se basa en una arquitectura de tipo serial. Sin embargo, es una interfaz de entrada/salida mucho más rápida que los puertos seriales estándar. La arquitectura serial se utilizó para este tipo de puerto por dos razones principales:

- La arquitectura serial le brinda al usuario una velocidad de reloj mucho más alta que la interfaz paralela debido a que este tipo de interfaz no admite frecuencias demasiado altas (en la arquitectura de alta velocidad, los bits que circulan por cada hilo llegan con retraso y esto produce errores);
- Los cables seriales resultan mucho más económicos que los cables paralelos.

1.6.2 Funcionamiento del USB

Una característica de la arquitectura USB es que puede proporcionar fuente de alimentación a los dispositivos con los que se conecta, con un límite máximo de 5 V por dispositivo. Para poder hacerlo, utiliza un cable que consta de cuatro hilos (la conexión a

tierra *GND*, la alimentación del *BUS* y dos hilos de datos llamados *D-* y *D+*) como se puede observar en la figura 1.9.

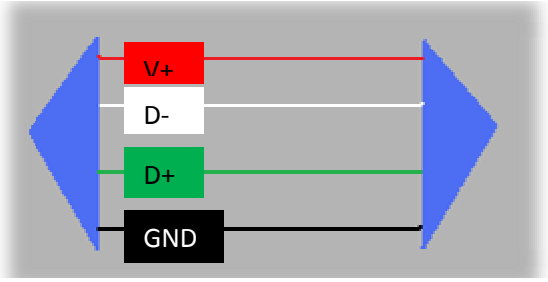


Figura 1.9 Hilos de conexión

Los puertos USB admiten dispositivos "Plug and play" de conexión en caliente. Por lo tanto, los dispositivos pueden conectarse sin apagar el equipo, de ahí el término conexión en caliente. Cuando un dispositivo está conectado al host, detecta cuando se está agregando un nuevo elemento gracias a un cambio de tensión entre los hilos D+ y D-. En ese momento, el equipo envía una señal de inicialización al dispositivo durante 10 ms para después suministrarle la corriente eléctrica mediante los hilos *GND* y *VBUS* (hasta 100 mA). A continuación, se le suministra corriente eléctrica al dispositivo y temporalmente se apodera de la dirección predeterminada (dirección 0). La siguiente etapa consiste en brindarle la dirección definitiva (éste es el procedimiento de *lista*). Para hacerlo, el equipo interroga a los dispositivos ya conectados para poder conocer sus direcciones y asigna una nueva, que lo identifica por retorno. Una vez que cuenta con todos los requisitos necesarios, el host puede cargar el driver adecuado.

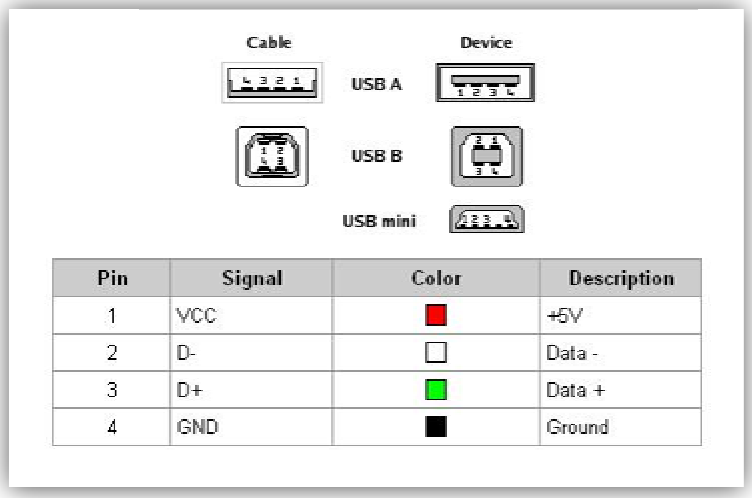
1.6.3 Tipos de conectores

Existen dos tipos de conectores USB:

- Los conectores conocidos como **tipo A**, cuya forma es rectangular y se utilizan, generalmente, para dispositivos que no requieren demasiado ancho de banda (como el teclado, el ratón, las cámaras Web, etc.);

- Los conectores conocidos como **tipo B** poseen una forma cuadrada y se utilizan principalmente para dispositivos de alta velocidad (discos duros externos, etc.).

Los 2 tipos de conectores se muestran en la figura 1.10



Pin	Signal	Color	Description
1	VCC		+5V
2	D-		Data -
3	D+		Data +
4	GND		Ground

Figura 1.10 Configuración de Conectores USB

1.6.4 Terminología USB

Host: Dispositivo maestro que inicia la comunicación (Generalmente la computadora).

Hub: Dispositivo que contiene uno o más conectores o conexiones internas hacia otros dispositivos USB, el cual habilita la comunicación entre el host y diversos dispositivos. Cada conector representa un puerto USB.

Dispositivo compuesto: Es aquel dispositivo con múltiples interfaces independientes. Cada una tiene una dirección sobre el bus pera cada interface puede tener un diferente *driver device* en el host.

Puerto USB: Cada host soporta solo un bus, cada conector en el bus representa un puerto USB por lo tanto sobre el bus puede haber varios conectores, pero solo existe una ruta y solo un dispositivo puede transmitir información a la vez.

Driver: es un programa que habilita aplicaciones para poderse comunicar con el dispositivo. Cada dispositivo sobre el bus debe tener un driver, algunos periféricos utilizan los drivers que trae Windows.

Puntos terminales (*Endpoints*): Es una localidad específica dentro del dispositivo. El *Endpoint* es un buffer que almacena múltiples bytes, típicamente es un bloque de la memoria de datos o un registro dentro del microcontrolador. Todos los dispositivos deben soportar el punto terminal 0. Este punto terminal es el que recibe todo el control y la peticiones de estado durante la enumeración cuando el dispositivo esta sobre el bus.

1.7 Procesamiento de imágenes y Visión computacional

La visión por computadora, dentro del campo de la computación inteligente, puede considerarse como el conjunto de todas las técnicas y modelos que no permiten el procesamiento, análisis y explicación de cualquier tipo de información espacial obtenida a través de imágenes digitales.

Debido a que la información visual es una de las principales fuentes de datos del mundo real, resulta útil el proveer a una computadora digital del sentido de la vista(a partir de imágenes tomadas con cámaras digitales), que junto con otros mecanismo como el aprendizaje hagan de esta una herramienta capaz de detectar y ubicar objetos en el mundo real, objetivo principal de la visión por computadora.

La siguiente área importante para la realización de este proyecto es el área de procesamiento de imágenes, la cual se combina con algoritmos para controlar el envió de datos por el puerto USB con el objetivo de poder controlar al brazo manipulador para que seleccione los objetos por su forma de manera correcta. Se puede decir que la visión computacional y el procesamiento de imágenes son dos campos que tienen mucho en común, pero el objetivo final es diferente. Para el caso del *procesamiento de imágenes* su objetivo final es la de mejorar la calidad de las imágenes para su posterior utilización e interpretación (Figura 1.11), diferenciándose de la *visión computacional* en que esta misma utiliza el procesamiento de imágenes para extraer características de una imagen para su descripción e interpretación por la computadora.

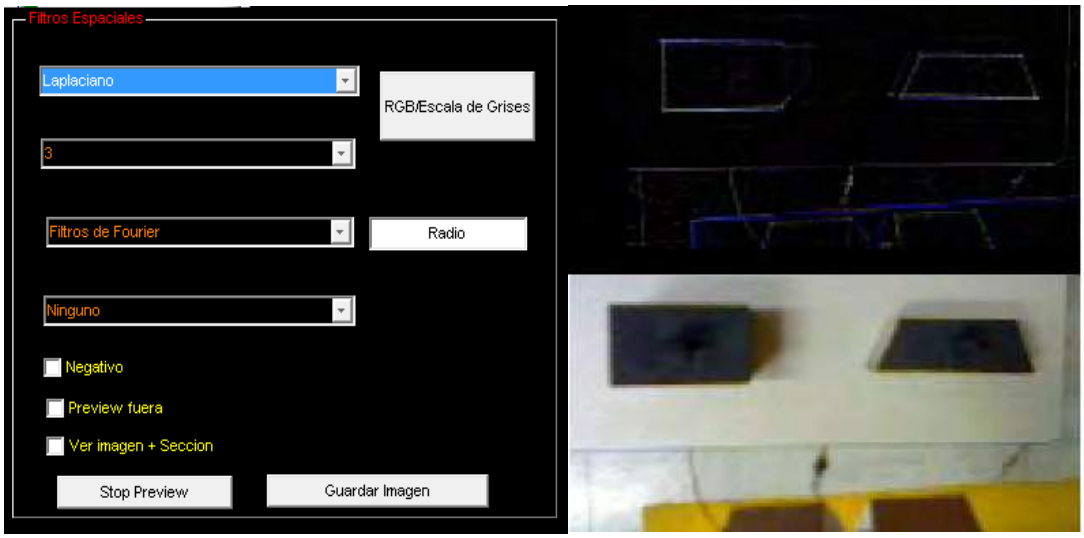


Figura 1.11 Pruebas en laboratorio de visión por computador y procesamiento de imágenes en tiempo real enfocados hacia la detección de forma, donde la cámara se encuentra en el manipulador

La base del software de un sistema de visión es la interpretación y análisis de los pixeles. El resultado final puede ser, desde la medida de una partícula, hasta la determinación o lectura de formas y colores de algún objeto pasando por algún procesamiento de imágenes. Cabe señalar también que dependiendo si la aplicación se realiza en un medio industrial o científico los pasos a seguir en un sistema de visión serán distintos.

Aplicación Industrial

- Captura de imagen
- Definición de la región de interés donde se realizara el reconocimiento
- Inicialización de los limites para determinar si la pieza a analizar es o no la correcta
- Ejecutar las mediciones o reconocimiento
- Generar una salida aplicada

Aplicación Científica

- Capturar imagen
- Hacer un proceso de mejora de la imagen
- Determinar los elementos a medir o reconocer

- Ejecutar la medición o el reconocimiento
- Analizar las medidas y realizar procesos gráficos o estadísticos

Mientras que en las aplicaciones industriales la velocidad a la que se realiza las medidas es fundamental, ya que se deben evaluar todas las piezas producidas en tiempo real, en las aplicaciones científicas se busca la determinación de los resultados en imágenes más complejas para poder desarrollar un sistema que sea más funcional.

1.7.1 Componentes de un sistema de visión

El buen funcionamiento de un sistema de visión depende en gran medida de sus componentes que lo forman, siendo 6 partes fundamentales para que el sistema llegue a un funcionamiento correcto, las cuales se mencionan a continuación:

- **Captación :** Es el proceso a través del cual se obtiene una imagen visual
- **Preprocesamiento:** Incluye técnicas tales como el rellenado hoyos, eliminación de ruido, borrado de objetos en los bordes además del realce de detalles.
- **Segmentación:** Es el proceso por el cual se selecciona parte de la imagen en donde se encuentra solamente el objetos de interés.
- **Descripción:** Es el proceso mediante el cual se obtienen las características convenientes del objeto a analizar, con el objetivo de poder diferenciar un tipo de otro, como puede ser forma, color, área, medidas de los ejes.
- **Reconocimiento:** Es el procedimiento por medio del cual se asocia un significado a un conjunto de objetos reconocidos

Todos estos componentes definidos se pueden observar en la figura 1.12 con un diagrama, en el cual se hace la separación entre hardware y software.

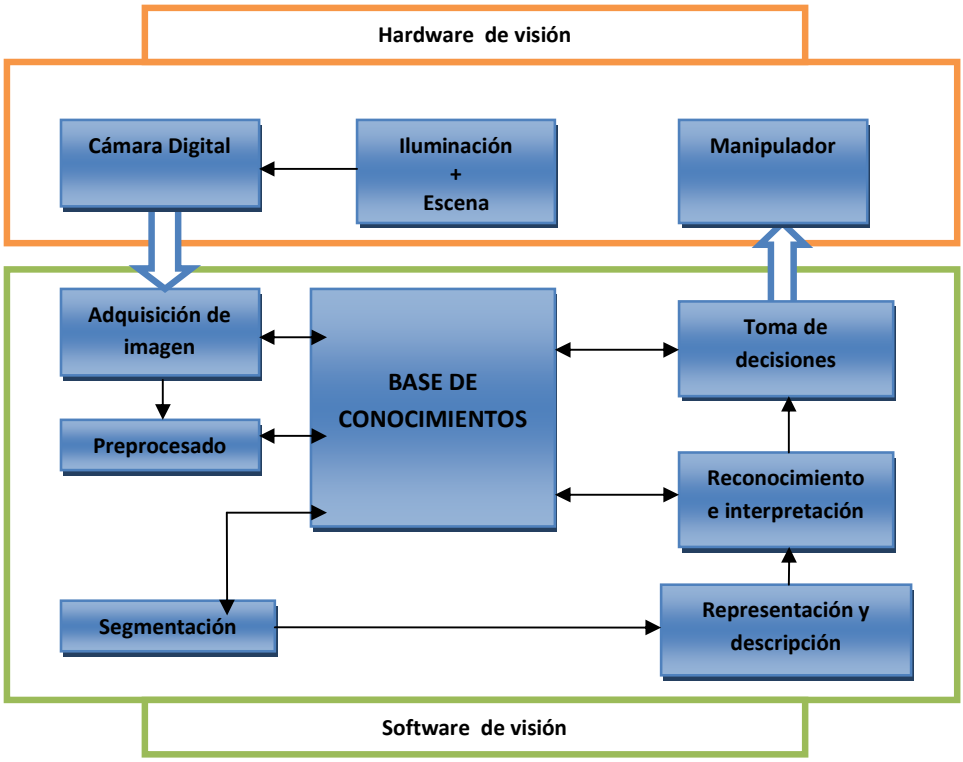


Figura 1.12 Diagrama de etapas fundamentales de un sistema de visión artificial

1.7.2 Utilidades de Robots y visión artificial

Los sistemas de visión artificial integrados en células robotizadas se utilizan con gran éxito para las siguientes utilidades:

- ***Medición:*** Mide las dimensiones de una pieza, sus diámetros, la planitud de superficies, niveles.
- ***Guiado:*** Guía a los robots para localizar o ensamblar piezas, guía el recorrido de un rollo de papel, tela, cartón.
- ***Identificación:*** Identifica piezas o productos por su perfil, realiza reconocimiento óptico de caracteres OCR.
- ***Inspección:*** Detecta la presencia-ausencia de piezas en ensamblaje, realiza la orientación de piezas, las inspecciones de defectos superficiales, la comprobación de la fecha en un lote de productos o su caducidad, los defectos en la superficie en productos producidos de forma continua.

En las figuras 1.13 y 1.14 se muestran los primeros resultados obtenidos con la unión del hardware y software de visión.

Figura 1.13 Ejemplo de identificación de forma rectangular

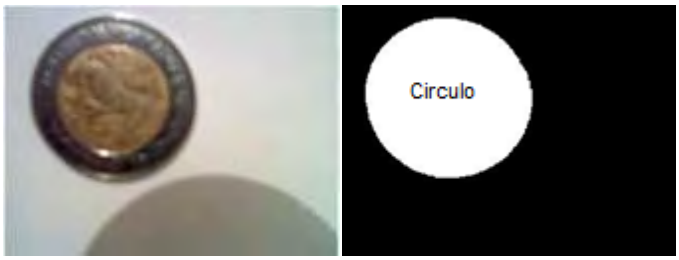


Figura 1.14 Ejemplo de identificación de forma circular

Además los sistemas de visión artificial contribuyen a los Controles de Calidad en los procesos productivos; procesos de verificación, mediante inspecciones 100%, evitando posibles escapes de la inspección ocular, lo cual garantiza la calidad y la homogeneidad de las piezas inspeccionadas.

CAPITULO 2

Comunicación del dispositivo de detección de forma con el Robot manipulador

2.1 Introducción

En el presente capitulo se explica como se construyo la interfaz para la comunicación entre el dispositivo de detección de forma y la unidad de control del KUKA-KR16, dicha interfaz tiene como objetivo principal el codificar datos de salida además de acoplar dos etapas de distinto voltaje y así lograr que la unidad de control con entradas digitales de +24v puedan sincronizarse y recibir las señales provenientes de puerto USB el cual es manipulado por el algoritmo de detección de forma realizado MATLAB. Esta interfaz consta de varias etapas que describen a continuación.

2.2 Emulación del puerto serial RS-232 con la clase USB CDC

El puerto serial RS-232 ya no es tan común en las nuevas generaciones de computadoras personales (PC). Debido a esto surgen grandes problemas para la comunicación con dispositivos externos y diversas aplicaciones desarrolladas y configuradas para este tipo de puerto. La solución está en emigrar la aplicación hacia el puerto USB.

Hay muchas diferentes maneras de convertir de puerto RS-232 a puerto USB, la más simple de ellas es la de emular un puerto serial sobre el bus de USB. Una ventaja de este método es que la PC reconocerá la conexión USB como un puerto RS-232 COM y por eso no se requieren cambios en la aplicación ya establecida. Otra ventaja que se puede mencionar de este método es que utiliza un driver de Windows incluido en las versiones de Windows 98 y versiones posteriores hasta llegar al Windows vista en el cual se tiene que descargar el driver especial para esta versión, como es el caso de este proyecto, haciendo innecesario el desarrollo de alguno de ellos.

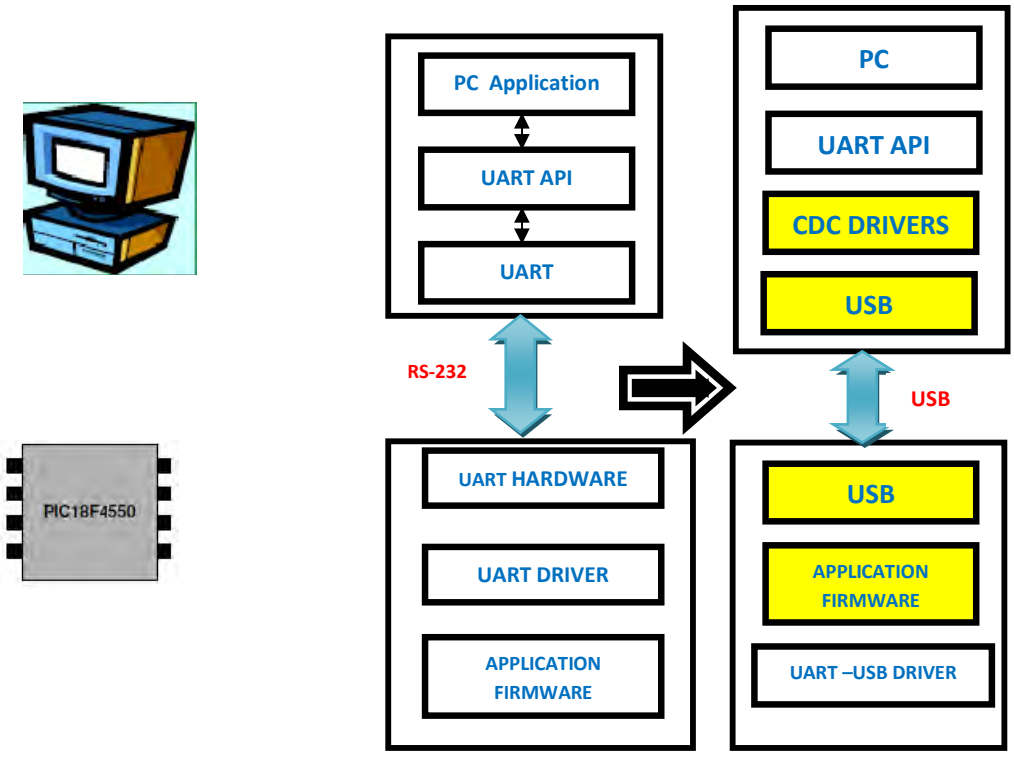


Figura 2.1 Cambios necesario para emigrar una aplicación de puerto RS-232 a USB

2.2.1 Sistema Incrustado (*Embedded System*)

Es un sistema computacional diseñado para realizar una o algunas funciones especializadas. En algunas ocasiones este tipo de sistemas son incrustados (*Embedded*) como una parte de un sistema completo que incluye hardware y partes mecánicas. En contraste a una computadora de propósito general tal como una computadora personal (PC) que es diseñada para ser flexible y para poder realizar diversas tareas.

Los sistemas incrustados son sistemas controlados por uno o más procesadores que son típicamente microcontroladores o procesadores digitales de señales (DSP).La característica principal sin embargo sigue siendo que son construidos para una aplicación en particular lo cual puede requerir procesadores muy poderosos. Un ejemplo puede ser un Windows *embedded* destinado al manejo de un Robot manipulador que a su vez es controlado por entradas y salidas digitales provenientes de un microcontrolador como es el caso en este proyecto.

Figura 2.2 Ejemplo de Windows Embedded dedicado para una aplicación en especifico

2.3 DETECCION DEL PUERTO SERIAL VIRTUAL

La clase del dispositivo de comunicación (CDC *Comunication device class*) define muchos modelos de comunicación incluyendo la emulación del puerto serial. El driver de Windows usbser.sys forma parte de dicha especificación. Por lo que "*Embedded device*" (modulo de comunicación) debe ser asignado en el driver instalado para que a la hora de la conexión pueda ser detectado y visualizado en el panel de control Windows como en la Figura 2.3.

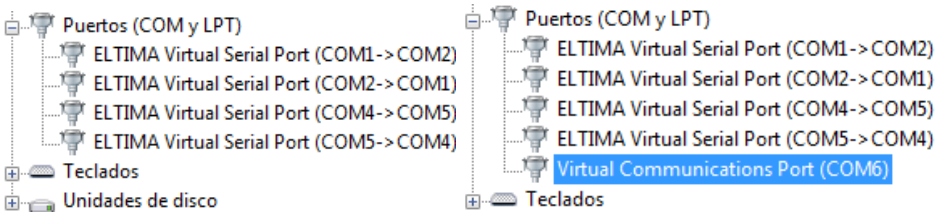


Figura 2.3 Visualización de la detección del dispositivo externo con un puerto serial virtual asignado por el sistema operativo

2.4 FUNCIONAMIENTO USB CDC

En resumen la especificación CDC requiere 2 interfaces. La primera es interface de la clase de comunicación (*Comunication class interface*), la cual utiliza un *Endpoint* de interrupción *IN*. Esta interface es usada para notificar al USB host de el estado de la conexión actual de RS-232 de el dispositivo emulado USB RS-232. La segunda es la interfaz de la clase de datos(*Data class Interface*),en la que se emplea un "OUT BULK *Endpoint*" y un "*IN BULK Endpoint* " la cual es usada para transferir los bytes de datos RAW que serian normalmente transferidos a través de el puerto serial RS-232 real.

2.4.1 FIRMWARE USB CDC

En electrónica e informática, el **firmware** es un término comúnmente utilizado para referirse a los servicios fijos, por lo general pequeños programas y estructuras de datos para el control interno de varios dispositivos electrónicos. Algunos ejemplos típicos de dispositivos que contengan firmware son los productos de usuario final, tales como calculadoras, *PLC's*, microcontroladores así como partes de computadoras y dispositivos como discos duros, tarjetas de memoria, todo enfocado hacia la instrumentación científica y robótica industrial. También los dispositivos más complejos, tales como teléfonos móviles, cámaras digitales, sintetizadores, contienen firmware.

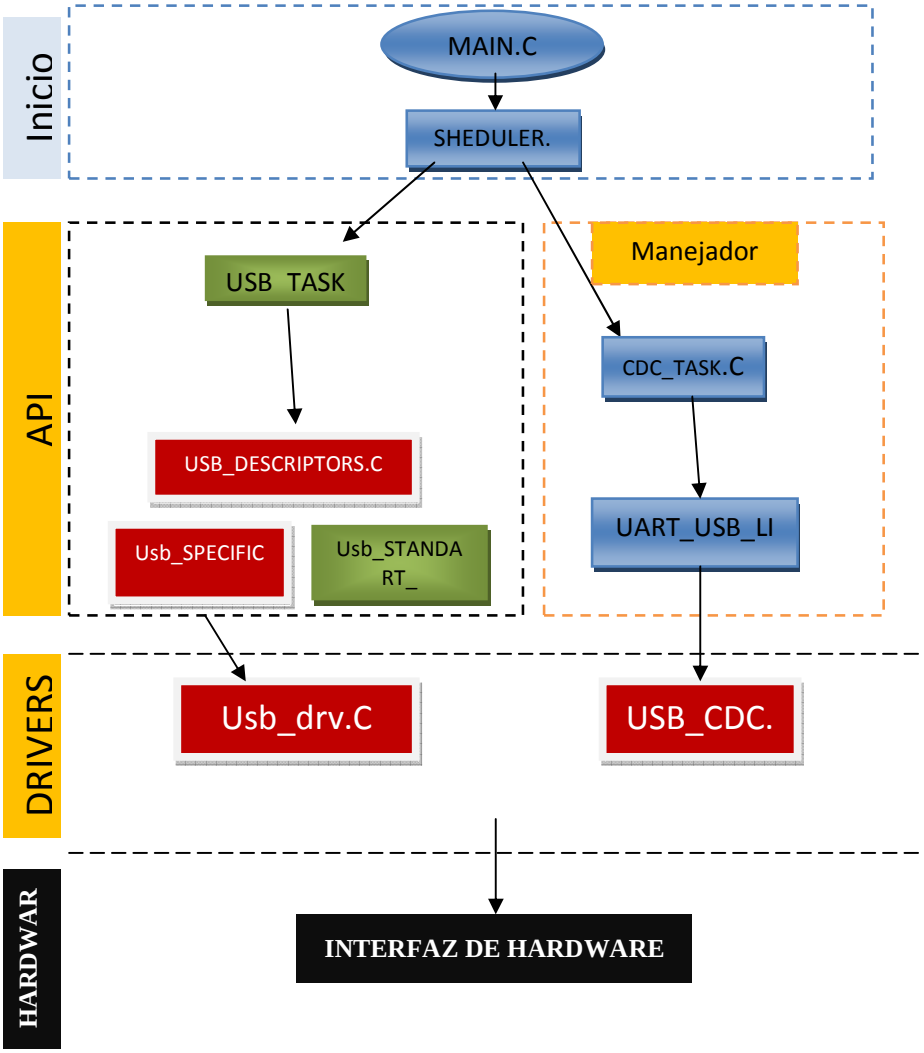


Figura 2.4 Arquitectura del FIRMWARE

Para el caso particular del firmware cargado en el microcontrolador PIC18F2550 utilizado para codificar toda la información proveniente del puerto USB, los pasos para que dicho microcontrolador detecte si la comunicación es en modo USB o USB CDC se muestran en la figura 2.3 y son explicados a continuación:

- **INICIO**

Cuando se conecta el modulo de comunicación en el cual está ubicado el microcontrolador que a su vez es cargado con las librería USB CDC, inicia a correr el programa principal (*main*) el cual detecta a dicha librería y así se puede activar los manejadores para dicho protocolo de comunicación.

- **API**

Una **interfaz de programación de aplicaciones (API)** es una interfaz implementada por un programa que le permite interactuar con otro software. Facilita la interacción entre los diferentes tipos de programas de forma parecida a la interfaz de usuario, facilitando la interacción entre humanos y computadoras. Una API es implementado por las aplicaciones, bibliotecas y sistemas operativos para determinar su vocabulario (USB CDC) y las convenciones de llamada, y se utiliza para acceder a sus servicios. Se pueden incluir las especificaciones, las rutinas, estructuras de datos, clases de objetos y los protocolos utilizados para la comunicación entre el consumidor y el ejecutor API.

- **DRIVER**

El driver para esta aplicación (Archivo con extensión .INF) es el que habilita aplicaciones de la PC para que se pueda comunicar con el dispositivo externo que en este caso es el microcontrolador PIC18F2550. Cada dispositivo sobre el bus debe tener un driver, algunos periféricos utilizan los drivers que trae Windows pero para este caso en particular se instalo el driver especifico para la comunicación USB CDC que fuera compatible con el sistema operativo Windows vista Home Premium.

Cuando se instala correctamente el driver compatible se puede visualizar en el administrador de dispositivos (Figura 2.5) el numero de puerto virtual asignado (que en

este caso es el COM 6) el cual puede variar según la entrada USB de la computadora donde se conecte la interfaz, mostrar si la conexión está funcionando correctamente, así como presentar la dirección asignada por el HUB en el bus.

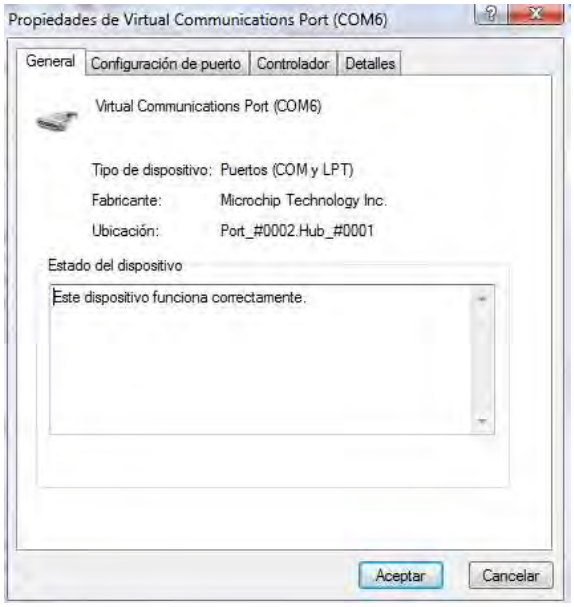


Figura 2.5 Ventana de Windows que muestra el estado de la conexión

2.5 INTERFAZ USB- KUKA KR16

Indicador de encendido

Cuando la extensión USB se conecta en la PC se activa un voltaje de 5v en la interfaz y el programa empieza a funcionar en un modo en el cual emite un sonido intermitente desde uno de los 4 relevadores, siguiendo así hasta que la interfaz de usuario (desde la PC en el botón seleccionar puerto COM Virtual) se sincroniza con la interfaz USB. Todos los circuitos de control de esta interfaz USB-KUKA KR16 se alimentan del voltaje suministrado por el puerto USB debido a que el circuito de control y acoplamientos requiere menos de 100 *mA* de corriente (esto es la cantidad máxima de consumo de corriente permitido en un solo puerto USB). La alternativa si se requiere utilizar circuitos externos es utilizar un regulador de voltaje estándar, como puede ser el LM7805 , el cual proporciona +5 V a 1000mA.

Para poder sincronizar la interfaz USB con el programa de reconocimiento se debe insertar el número de puerto virtual (observar figura 2.6) que se asigna automáticamente por el sistema operativo cuando se conecta dicha tarjeta. En el botón con el nombre “Puerto COM Virtual” se le inserta el número mencionado de puerto para poder establecer la comunicación entre la PC y la tarjeta electrónica que se fabrico para tal fin.

Figura 2.6 Configuración para el modo CDC desde el programa de reconocimiento de forma con el microcontrolador PIC

Conector USB

El conector USB es un conector estándar de "tipo A". Al cual llega la extensión de USB proveniente del puerto de la PC. Como se menciono antes existen cuatro hilos de conexión de un cable USB, de dos de ellos proviene la fuente de alimentación de la placa, mientras que los otros dos son las líneas de comunicaciones D + y D-. De esta manera es como se transfiere la información entre la PC y el PIC cuando la PC envía el código de la figura detectada.

Puertos de salida del PIC

Los puertos de salida del PIC utilizados para las salidas digitales que van hacia las entradas del manipulador se señalan en la figura 2.7, dichos puertos configurados como de salida son PB0, PB1, PB2 y PB7 correspondientes cada uno a las 4 figuras que se tiene programado reconocer (Triangulo, circulo, cuadrado y rectángulo).

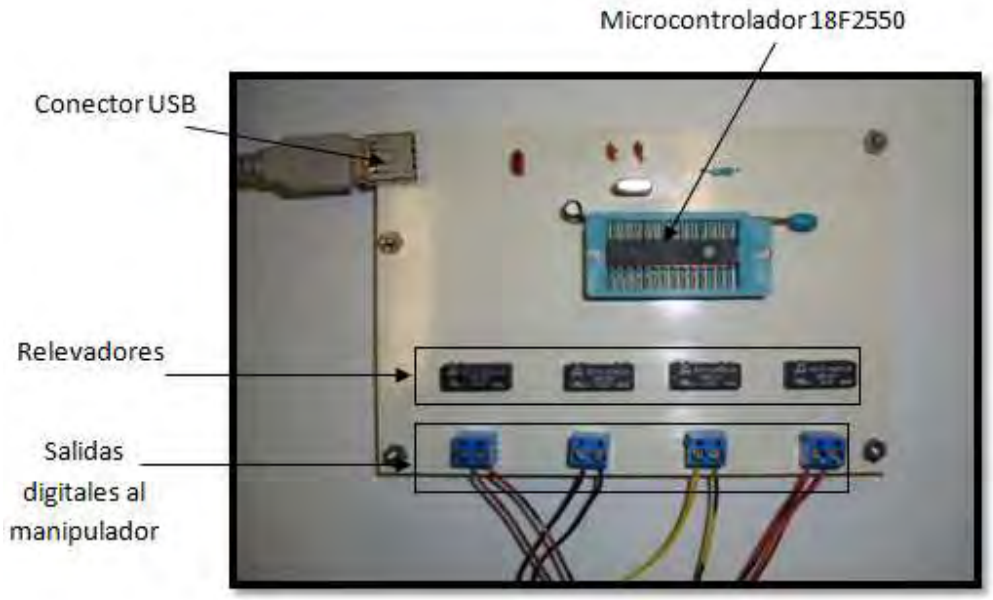


Figura 2.7 Interfaz USB-ROBOT KUKA KR16 con 4 salidas digitales

PIC18F2550

El PIC el cual se ubica en una base casi al centro del tablero. Es un microcontrolador programable con 32Kbytes de memoria de programa y una RAM de 2kBytes de propósito general. Cuenta con 10 canales A / D y 3 puertos cada uno con 8 entradas o salidas según se configure. Además de contar con sus terminales especificas para la conexión de los cuatro hilos desde el puerto USB

Oscilador de cristal de 12 MHz

Esta parte tiene la tarea de proporcionar una señal de reloj al PIC, y sus correspondientes capacitores de 10pF y su resistencia de 1KOhm proporciona la frecuencia adecuada para el microcontrolador.

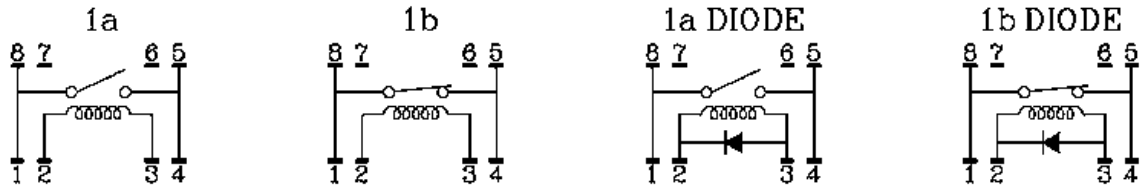
Relevadores

Un **Relevador,** también conocido como *relay*, es un dispositivo que controla el estado de un interruptor mediante una entrada eléctrica, para este caso en particular la dicha entrada eléctrica es la señal (0 y +5v) generada por el PIC. En su interior, posee comúnmente una bobina que al energizarse (por Ley de *Faraday*) induce una fuerza magnética que cambia el estado de dicho interruptor.

Existen relevadores con interruptores normalmente abiertos (es decir sin flujo eléctrico) los cuales son usados para esta aplicación en particular y se muestran en la figura 2.8, además de existir los normalmente cerrados. Además de esa característica también existen relevadores con múltiples entradas y múltiples interruptores.

a la entrada

La composición interna de este relevador donde se puede observar la bobina que atrae al interruptor que cambia de estado cuando llega la señal eléctrica del PIC, los 2 se muestran en la figura 2.9.



su diferentes

Salidas digitales

Un módulo programable con salidas digitales permite actuar sobre los preaccionadores y accionadores de algún sistema con entradas digitales como el modulo de E/S del KUKA KR16 que admitan ordenes en el modo de “conduce- no conduce”.

El valor binario de las salidas digitales se convierte en la apertura o cierre de algún relevador de la tarjeta como se puede observar en la figura 2.9. En los módulos estáticos, los elementos que conmutan son los componentes electrónicos como transistores o triacs, y en los módulos electromecánicos son contactos de relés internos al módulo.

Los módulos de salidas estáticos al suministrar tensión, solo pueden actuar sobre elementos que trabajan todos a la misma tensión, en cambio los módulos de salida electromecánicos, al ser libres de tensión, pueden actuar sobre elementos que trabajen a tensiones distintas como en el caso de este proyecto que las salidas del la interfaz son de 0 y 5V y las entradas del modulo de E/S son de 0 y +24v.

Después de definir todos los elementos que componen a esta interfaz, se muestra el diagrama completo de conexión de todos ellos en la figura 2.10.

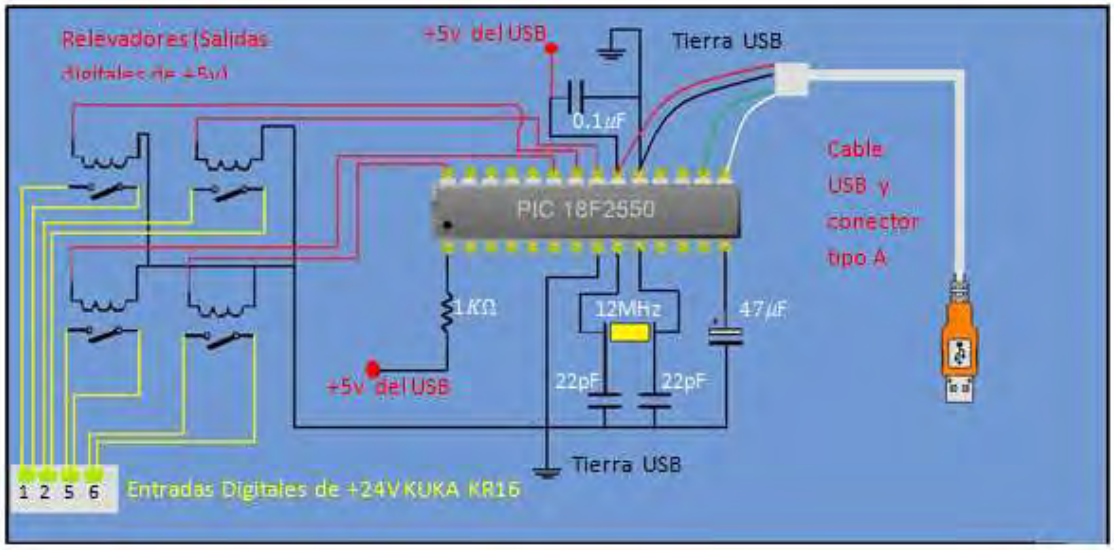


Figura 2.10 Diagrama completo de la interfaz USB-KUKA KR16

Entradas digitales

Los módulos de entrada digitales permiten conectar el robot, con los sensores que envían señales digitales, como sensores de fin de carrera, presencia, contacto, etc. Los módulos de entrada digitales trabajan con señales de voltaje, por ejemplo cuando por una entrada llegan 24 volts se interpreta como un "1" y cuando llegan cero volts se interpreta como un "0".

Las entradas digitales de +24v del modulo de E/S WAGO(1, 2, 5 y 6), encargadas cada una de ellas de captar la señal enviada por la interfaz USB-KUKA la cual activa una de ellas a la vez dependiendo del tipo de figura que sea detectada, se pueden observar a continuación en la figura 2.9 marcadas con letra blanca.

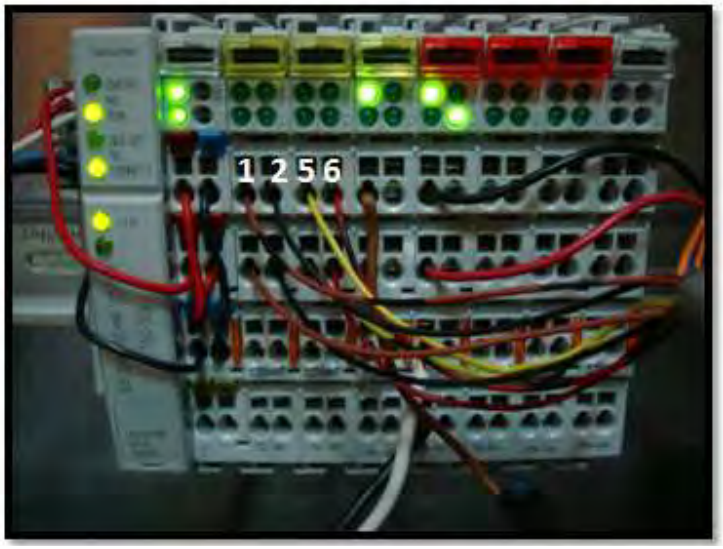


Figura 2.11 Entradas Digitales al KUKA KR16 por medio del modulo de E/S WAGO

Como se puede observar en la figura el modulo cuenta con 12 entradas (amarillas) de las cuales se utilizan 4 y 12 salidas (rojas) de las que 2 son usadas para controlar la apertura o cierre del gripper así como para alimentar el sensor del sistema anticolisión que se encuentran en la muñeca del manipulador.

Para poder enviar los datos de la figura que fue detectada después de ejecutar el algoritmo de reconocimiento de forma, se presiona el botón “Enviar pieza” con el numero 9, tal como se muestra en la figura 2.12. Este botón activa la función que envía el caractér

identificador de figura detectada, el cual fue programado en MATLAB para manejarlos de forma serial, pero siendo enviados por el puerto USB.

Figura 2.12 Botón enviar pieza

2.6 Conclusiones del capitulo

Al terminar la interfaz de comunicación se logro establecer una sincronización entre el programa de reconocimiento y el manipulador KUKA KR16, lo que permitió que el manipulador pudiera recibir, aceptar y entender las señales codificadas correspondientes a cada figura detectada, lo cual dio lugar a que las diferentes trayectorias del robot pudieran ser manejadas por el programa de reconocimiento, a causa de esto, se logra establecer uno de los objetivos más importantes de este proyecto, que consiste en que el manipulador sea manejado por software y no por un ser humano .

CAPITULO 3
Reconocimiento de forma

3.1 Introducción

Una parte fundamental para el desarrollo de este proyecto es el reconocimiento de forma, el cual se realiza por medio de un software de visión artificial programado en MATLAB y que tiene como objetivos principales:

- La **captura de imágenes** en tiempo real de la escena donde se encuentra la figura a reconocer, la cual se realiza por medio de una cámara ubicada en la muñeca del manipulador.
- La **toma de decisiones** a partir del algoritmo de reconocimiento de forma diseñado para este proyecto, lo que determinara la trayectoria o camino que el manipulador deba seguir.
- El manejo a nivel software de las señales necesarias para la **comunicación** con el manipulador.

En el presente capitulo se describen todos los pasos y consideraciones necesarias para realizar el dicho reconocimiento de forma y se describen a continuación.

3.2 Sistema de Visión artificial

El sistema que permite la captura y/o adquisición de la imagen, está formado por los elementos que se observan en la figura 3.1:

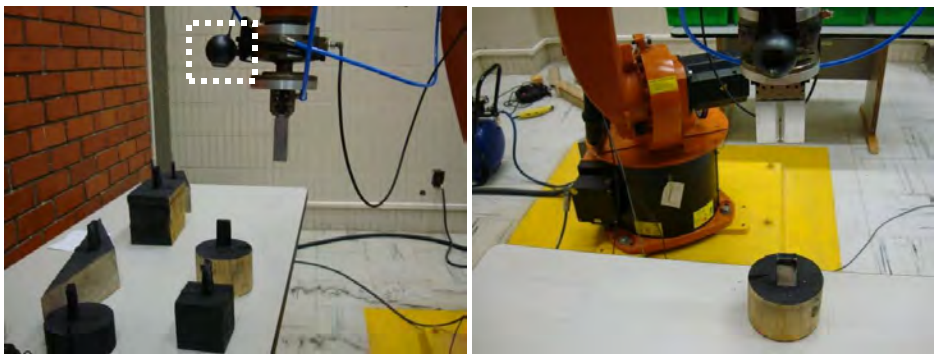


Figura 3.1 Webcam utilizada y su ubicación en la muñeca del manipulador

3.2.1 Cámara

Es el dispositivo encargado de transformar las señales luminosas que aparecen en la escena, en señales digitales capaces de ser transmitidas por un cable serial. Se divide en dos partes, el sensor, que captura las propiedades del objeto en forma de señales luminosas y lo transforma en señales digitales, y la óptica que se encarga de proyectar los elementos presentes en la escena a la superficie del sensor. En la figura 3.2 se muestra una imagen capturada en la que los parámetros de contraste y brillo fueron ajustados para obtener una imagen que sea más fácil de reconocer.

Figura 3.2 Imagen con alto contraste e Iluminación frontal

Los sensores de visión usados más recientemente son los basados en matrices de dispositivos acoplados por carga CCD; estos transductores proporcionan una señal con amplitud proporcional a la luminosidad de la escena y realizan una digitalización espacial completa en dos dimensiones (líneas y columnas), pues descomponen la imagen en una matriz de puntos.

Una vez que se está ejecutando el programa descrito en el capítulo 2 se selecciona la cámara que se va a utilizar para adquirir la imagen, primero se presiona el botón "seleccionar cámara" (Figura 3.2), posteriormente se activa una nueva ventana en la que se podrá seleccionar la cámara a usar, en la que la resolución programada para trabajar es 320x240 debido a que disminuye el tiempo de procesamiento y una resolución alta no es

tan necesaria para esta aplicación debido a que el enfoque está en la forma de los objetos a partir del contraste y no en sus detalles .

Figura 3.2 Botón para seleccionar cámara

3.2.2 Sistema de iluminación

La iluminación de la escena es un aspecto decisivo en el desarrollo de un sistema visual. Antes de intentar corregir un problema de iluminación por medio de algoritmos muy complicados, es mejor prestar atención e implementar un sistema de iluminación adecuado para que la captura de la imagen sea correcta. Es mejor un buen sistema de iluminación, que intentar corregir ese problema por software, pues la velocidad de procesamiento será mayor con algoritmos más sencillos. Por lo tanto, se menciona a continuación el sistema de iluminación usado para esta aplicación, exponiendo una breve descripción.

Iluminación Frontal

Es la más usada, y como se muestra en la figura 3.3 consiste en iluminar frontalmente la pieza, presenta más problemas para obtener un buen contraste entre la pieza y el fondo debido a la aparición de brillos y sombras que alteran las propiedades de las piezas a estudio lo que se puede evitar seleccionando colores contrastantes de pieza y superficie, además de pintar las piezas con pintura poco reflectora para evitar los brillos.

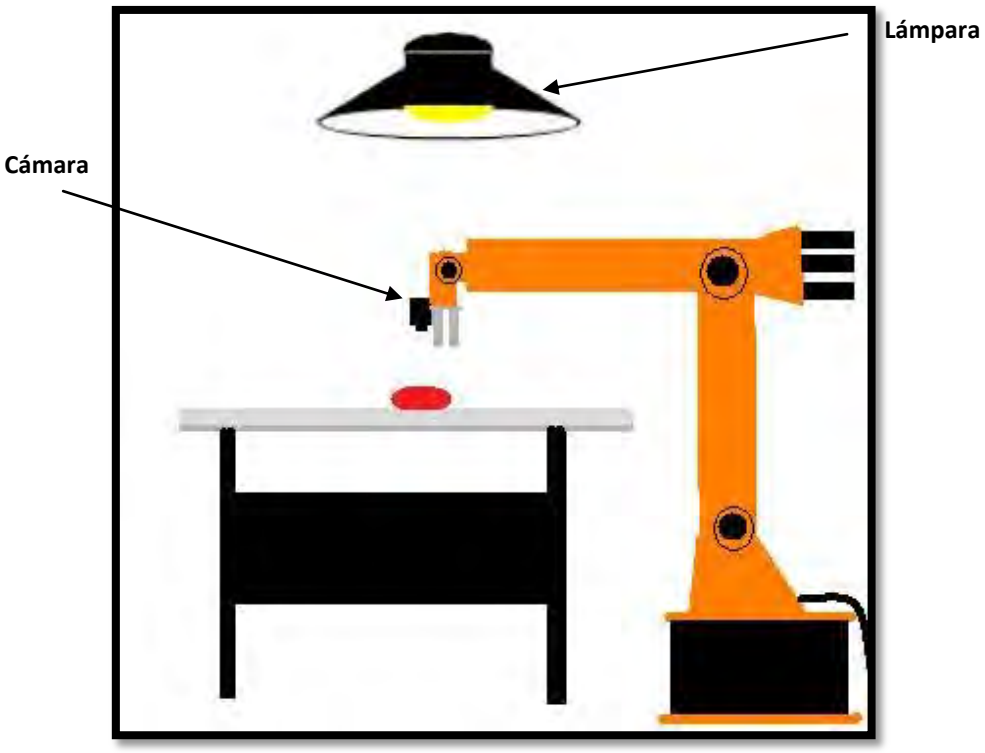


Figura 3.3 Iluminación Frontal

3.2.3 Fuente de iluminación

La fuente de iluminación usada para este sistema es una lámpara incandescente consiste en un filamento de tungsteno o halógeno-tungsteno. Como ventaja tiene que existe gran variedad de potencias y como desventaja, que reduce su luminosidad con el tiempo, lo que puede provocar problemas en algunos sistemas de visión.

3.3 PREPROCESAMIENTO DE IMAGEN

3.3 .1 Binarización

La mayor parte de los algoritmos para reconocer formas están escritos a partir de imágenes binarias, por lo que se hace conveniente el paso de una imagen en niveles de gris (o color) a una binaria, además esto permite reducir el volumen de los datos a tratar.

La binarización de una imagen digital consiste en convertir la imagen digital en una imagen en blanco y negro, de tal manera que se preserven las propiedades esenciales de la imagen, como se muestra en la figura 3.4.

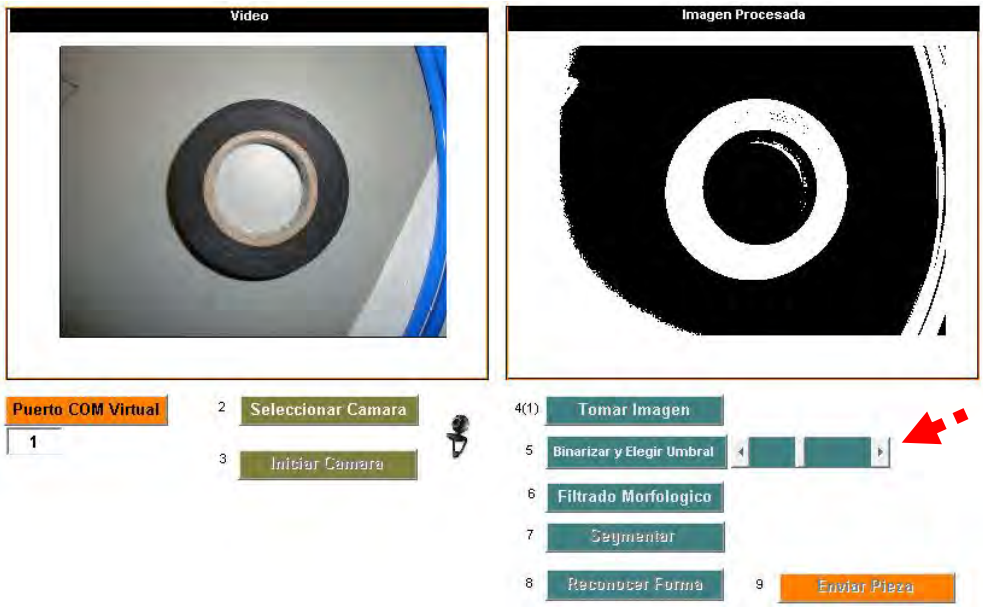


Figura 3.4 Botón para el primer paso del procesamiento de imágenes además de poder cambiar el umbral de binarizacion con la barra de la derecha

Uno de los métodos para poder binarizar una imagen digital es mediante el histograma de dicha imagen. A través del histograma obtenemos una gráfica donde se muestran el número de píxeles por cada nivel de gris que aparecen en la imagen. Para binarizar la imagen, se deberá elegir un valor adecuado dentro de los niveles de grises (umbral) con la barra de desplazamiento a la derecha del botón "*Binarizar",* de tal forma que el objeto a analizar quede bien definido hasta un cierto nivel donde no se noten algunos efectos no deseados como el de sombra que nos deforman el objeto que se quiere reconocer. Dicho efecto podría llevar a una detección errónea y por consiguiente, a la hora de la selección llegar a ubicar ese objeto detectado en el lugar que no le corresponde. Todos los niveles de grises menores al umbral calculado se convertirán en negro y todos los mayores en blanco.

3.3.2 Operaciones morfológicas

La morfología matemática es una herramienta muy utilizada en el procesamiento de imágenes. Las **operaciones morfológicas** pueden simplificar los datos de una imagen, preservar las características esenciales y eliminar aspectos irrelevantes. Teniendo en cuenta que la identificación y descomposición de objetos, la extracción de rasgos, la localización de defectos e incluso los defectos en líneas de ensamblaje están sumamente relacionados con las formas, es obvio el papel de la morfología matemática. La morfología matemática se puede usar, entre otros, con los siguientes objetivos:

- Preprocesamiento de imágenes (supresión de ruido, simplificación de formas, etc.).
- Destacar la estructura de objetos (extraer el esqueleto, marcado de objetos, envolvente convexa, ampliación, reducción, etc.).
- Descripción cualitativa de objetos (área, perímetro, diámetro, etc.).

Dilatación y Erosión

Estas operaciones son fundamentales en el procesamiento morfológico. De hecho, la mayoría de los algoritmos morfológicos están basados en estas dos operaciones aunque también se utilizan las siguientes:

• Opening and Closing (Apertura y cierre)
• Extracción de Frontera (Boundary Extraction)
• Afinado o Adelgazamiento (Thinning)
• Engrosamiento (Thickening)
• Relleno de Región (Region Filling)
• Esqueleto
• Poda (Pruning)
• Reflexión
• Resta de imágenes

Dilatación

Tomar cada píxel del objeto (con valor “1”) y cambiar a “1” todos aquellos píxeles pertenecientes al fondo (*background*) que tienen una conectividad alta con el píxel del objeto. En pocas palabras, poner a “1” los píxeles del fondo vecinos a los píxeles del objeto.

Erosión

Tomar cada píxel del objeto que tiene una conectividad C con los píxeles del fondo y cambiarlo a “0”. En otras palabras, poner a “0” los píxeles del objeto vecinos a los píxeles del fondo. Después de pasar por el proceso de binariazación se deben realizar algunas operaciones morfológicas para poder dejar bien definida el área a analizar y minimizar efectos como la sombra.

Eliminar áreas menores a 100 pixeles

Este paso que se puede visualizar en la figura 3.4, se ejecuta con la intención de eliminar objetos intrascendentes que pueden interferir a la hora de ejecutar el algoritmo de detección, esto se hace también con la finalidad de que el objeto a detectar quede totalmente aislado y no exista ningún otro objeto que lo pueda que pueda deformar la imagen.

Figura 3.5 Imagen con formas de menores de 100 pixeles sin eliminar y a la derecha dichas formas eliminadas

Eliminar objetos incompletos

Con esta operación se eliminan los objetos que se encuentran unidos al borde como triángulos, cuadrados, círculos y rectángulos incompletos, los cuales no tienen una forma definida que pudiera ser detectada para ser seleccionada como se puede observar en la figura 3.5.

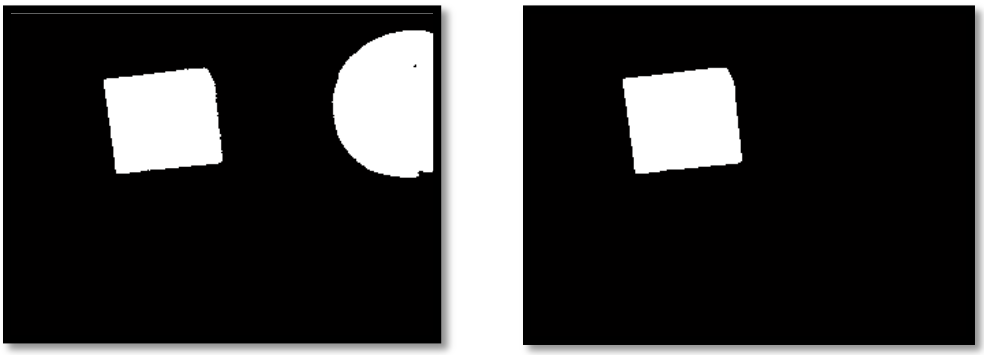


Figura 3.6 Circulo incompleto eliminado

Separar objetos

El proceso que se muestra la figura 3.7 desarrolla una apertura morfológica, esto quiere decir que separa dos objetos que se encuentren unidos por unos cuantos pixeles, primero ejecuta una erosión seguida de una dilatación.

Figura 3.7 Separación de imágenes unidas por algunos pixeles

Suavizar bordes

Para eliminar pixeles que se pueden visualizar como picos que se encuentren en el borde de las imágenes, se ejecuta una operación morfológica de cierre en la que primero se dilata la imagen y después se erosiona.

Rellenar agujeros

Debido a que este tipo de reconocimiento se realiza tomando en cuenta el área de las figuras, se rellenan agujeros que se encuentran en el interior de las formas a detectar, como se muestra en la figura 3.8.

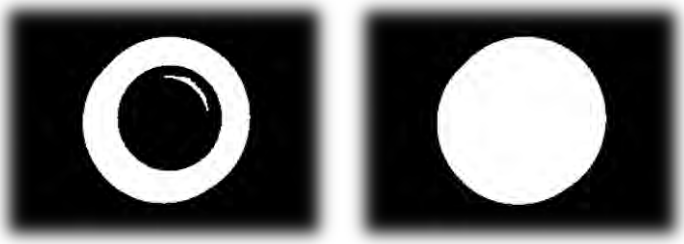


Figura 3.8 Rellenado de agujeros

3.4 Segmentación

La **segmentación** en el campo de la visión por computadora es el proceso de dividir una imagen digital en varias partes (grupos de pixeles) u objetos. El objetivo de la segmentación es simplificar y/o cambiar la representación de una imagen en otra más significativa y más fácil de analizar. La segmentación se usa tanto para localizar objetos de interés, como para encontrar los límites de estos dentro de una imagen (Figura 3.9). Más precisamente, la segmentación de la imagen es el proceso de asignación de una etiqueta a cada pixel de la imagen de forma que los pixeles que compartan la misma etiqueta también tendrán ciertas características visuales similares.

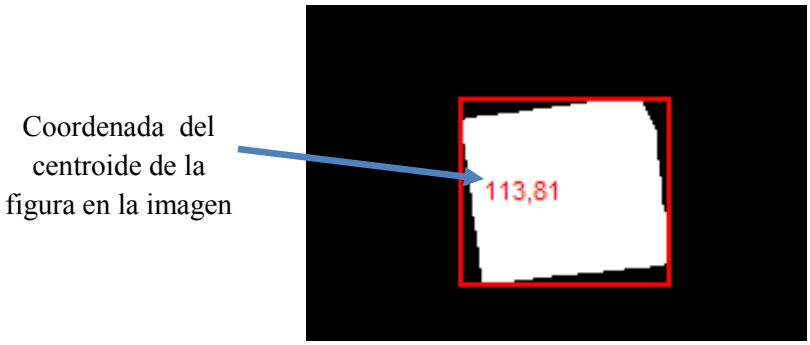


Figura 3.9 Región de interés segmentada

El resultado de la segmentación de una imagen es un conjunto de segmentos que cubren en conjunto a toda la imagen, o un conjunto de contornos extraídos de la imagen (véase la detección de bordes). Cada uno de los pixeles de una región son similares en alguna características, como el color, la intensidad o la textura. Regiones adyacentes son significativamente diferentes con respecto a la misma característica.

A través de propiedades básicas como: área, centroide, se puede definir una región dentro de una imagen. Dicho proceso el cual fue realizado en Matlab se puede ver con más detalle en el apéndice A en la sección de segmentación y etiquetación de la imagen.

La variable "*Img_Binarizada*" (en el apéndice A) es la imagen que contiene regiones que queremos describir. El primer paso que se realiza es convertir dicha imagen a una representación a través de una matriz etiquetada (*Labels*) como se muestra en la figura 3.10, usando para ello el segundo comando" *bwlabel*". La siguiente instrucción (en el apéndice A) toma como elemento de entrada a esa matriz etiquetada y calcula las propiedades necesarias como longitud de ejes, área y centroide.

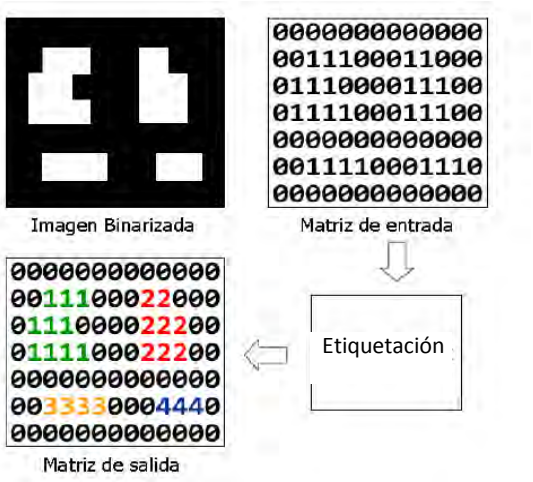


Figura 3.10 Matriz etiquetada

Si analizamos los resultados obtenidos podemos mencionar que el algoritmo de segmentación basado en etiquetación de píxeles propuesto resulta muy efectivo en condiciones en las cuales sabemos que los objetos a segmentar no están juntos en la escena.

3.5 Algoritmo de reconocimiento

Un método (para el caso del circulo y triangulo) de los dos utilizados para reconocer las cuatro formas contempladas para este proyecto es tomar decisiones a partir de un área determinada, la base de la diferenciación radica en que una figura tendrá un área dentro de un rango de valores y la otra figura otro rango diferente a partir de una referencia que se establece redimensionando todas las imágenes segmentadas (para este caso a 50x50 pixeles). El otro método (para el caso del cuadrado y rectángulo) radica en conocer la diferencia entre longitudes de sus ejes por medio de la función *regionprops*, dicho procedimiento se muestra en al apéndice A en la parte de reconocimiento

1- Redimensionar imagen a 50x50 pixeles

Para poder establecer una referencia de un rango de valores de área al momento de clasificar las formas, se redimensionan todas las imágenes que fueron resultado de la segmentación a 50x50 pixeles como se muestra en la figura 3.11.

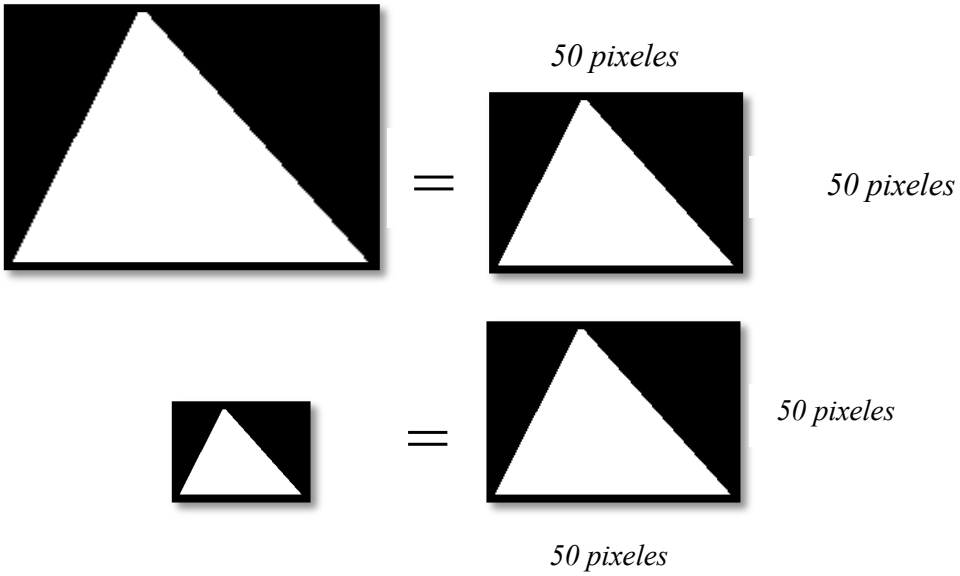


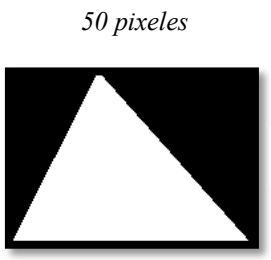
Figura 3.11 Imágenes segmentadas redimensionadas

2-Suma de pixeles blancos

Después de redimensionar la imagen se suman, y se almacenan en una variable los pixeles con valor de "1" y se descartan los pixeles en "0", determinando con esto el área total de la figura en pixeles, que después pasara a comparación para establecer de que figura geométrica se trata.

3-Determinacion del tipo de forma

Se calcula el rango máximo en pixeles que podría tener el área de un triangulo en las dimensiones de 50x50 como se muestra en la figura 3.12. Entonces el valor a comparar será el máximo posible.



$$\frac{BxA}{2} = \frac{50x50}{2} = 1250$$

Figura 3.12 Calculo del valor máximo del área para el triangulo

Entonces todas imágenes que tenga un área de pixeles blancos menor a 1250 serán detectados como triángulos, a su vez de manera similar se estableció como en la figura 3.13 el parámetro máximo para el circulo.

$$\pi * r^2 = 3.1416 * 25^2 = 1963.5$$

Figura 3.13 Calculo del valor máximo del área para el circulo

Si el área de la imagen es mayor a 1260 (umbral de 10 pixeles) y menor a 1963 entonces es un círculo.

Después de de haber comparado con los primeros dos parámetros anteriores de área establecidos, si no concuerda con ninguno de ellos es que es un cuadrado o un rectángulo y pasa a la siguiente comparación mostrada en la figura 3.14, la cual hace diferencia entre las dos figuras mencionadas mediante el siguiente procedimiento:

1.- Se guarda un valor mínimo de diferencia, por ejemplo 15 pixeles.

2.- Se obtiene el valor en pixeles de eje mayor.

3.- Se obtiene el valor en pixeles de eje menor.

4.- Se restan el valor del eje mayor menos el eje menor.

5.- Si la diferencia entre el eje mayor y eje menor es menor a 15 pixeles entonces se trata de un cuadrado, de lo contrario se trata de un rectángulo.

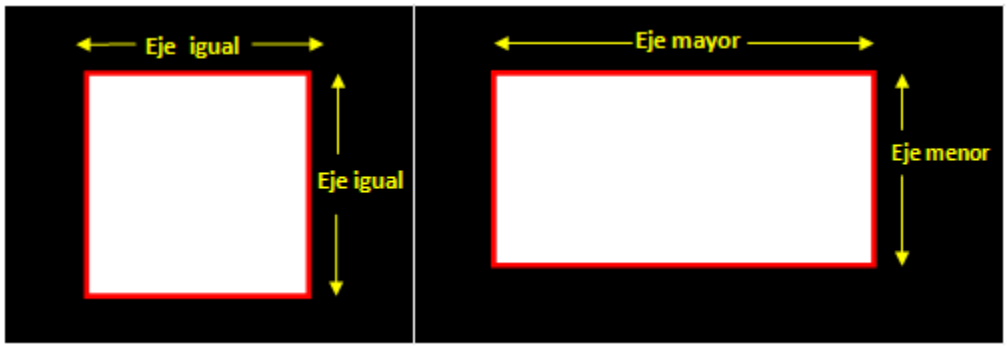


Figura 3.14 Ejes de Cuadrado y rectángulo

$$\boldsymbol{Eje\ mayor - Eje\ menor \leq 15 pixeles\ \therefore Es\ cuadrado}$$

$$\boldsymbol{Eje\ mayor - Eje\ menor > 15 pixeles\ \therefore Es\ Rectangulo}$$

3.6 Detección de forma en tiempo real

A diferencia del procedimiento para la detección de forma anterior, en este método se realiza la binarización de la imagen a partir de un color seleccionado, esto quiere decir que solo tomaran el valor de "1" los pixeles del mismo color seleccionado y "0" los demás pixeles que no coincidan con dicho parámetro ,como lo muestra figura 3.15, en donde se

puede observar que el sistema se ha calibrado para que determine la forma de objetos que solamente del color establecido previamente y que en la figura 3.15 son enmarcados **por el programa** con un con rectángulo de línea continua, ignorando por completo una figura circular de color diferente (enmarcada, solo para la imagen, en línea discontinua). Además para hacer que el proceso se continúe sin la necesidad de que algún operador este enviando la pieza, se automatizo la parte del envió del ID identificador de cada figura (lo cual también se hacía manualmente en el procedimiento del apartado 3.2), el retardo en este envió depende de la cantidad de cuadros capturados y se puede modificar dependiendo del tiempo en el que el manipulador realice la trayectoria para depositar la figura seleccionada.

Figura 3.15 Detección de forma tomando como referencia el color

3.6.1 Tomar imagen y seleccionar color

Tomar Imagen

El primer botón "Tomar Imagen" se utiliza para detener el video y capturar un cuadro 320x240 estático en el cual se seleccionara el color.

Seleccionar Color

De la matriz (Imagen completa) del paso anterior se extrae una parte de ella la cual es una matriz RGB más pequeña que es mayormente la tonalidad que se quiere seleccionar, a la cual se le separan los tres canales y se cambia a vector cada uno de ellos para sacarle el promedio, lo que da por resultado tres valores correspondientes a cada canal, después se juntan para obtener el color al cual se tomara por referencia. El color seleccionado aparece en un pequeño cuadro en la figura 3.16 a lado derecho del botón de "seleccionar color".

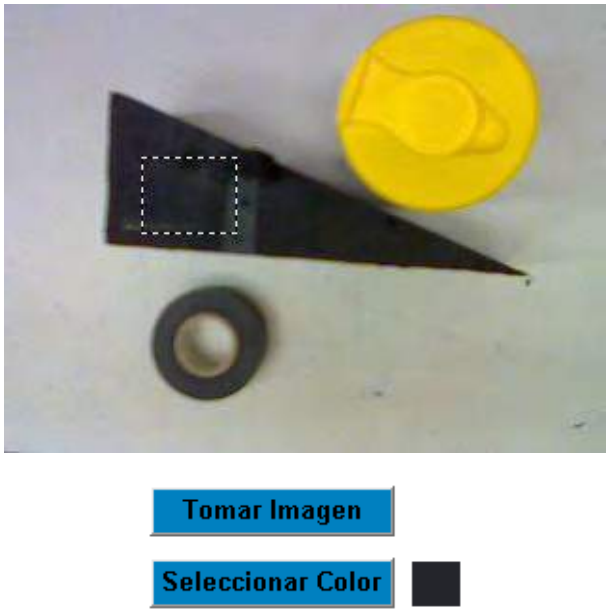


Figura 3. 16 Selección del color a partir de la imagen real

Después de que se selecciona el color, se exporta a otra función para ser utilizado como medio de comparación para el siguiente paso en el que ya se empieza a detectar la forma de la figura y es descrito en el siguiente apartado.

Para explicar más claramente la manera en que se obtiene dicha referencia, se estableció el siguiente procedimiento:

1.- Se genera una matriz sencilla de 2x2 simulando que es un canal de la matriz RGB de la parte extraída en la imagen original.

2.- El siguiente paso es convertir esta matriz a un vector:

3.- Para obtener el valor final de un canal se obtiene el promedio de este vector aproximándolo al entero más próximo y se hace lo mismo para los 3 canales.

3.6.2 Seguir forma

Al presionar el botón “Seguir forma”, inicia un ciclo infinito que solo puede ser detenido por el botón “Detener Detección”, en el cual se va procesando cuadro por cuadro como si se tratara de imágenes capturadas a mano en el proceso de la sección 3.2 el cual no es automático, pero antes de aplicar el algoritmo de detección se debe binarizar la imagen por un método diferente al de la detección del apartado 3.2, pues la calibración se hace mediante el color y la binarización, a diferencia del método que no es automático en el que solo se modifica el umbral para binarizar la imagen.

3.6.2.1 Binarización a partir de un color determinado

Una vez que inicia el ciclo infinito, el primer paso que se ejecuta es la binarizacion, la cual como se menciono, es a partir de un color seleccionado de uno de los cuadros de la señal de video antes de empezar el seguimiento de forma; como se puede observar en la figura 3.17, se eligió un color muy aproximado al negro (color original de las piezas a detectar), para poderlo explicar con más claridad, se detiene la detección de forma en tiempo real que se encuentra ejecutándose en ese momento, después se guarda un cuadro del video en una variable del WorkSpace (MatLab) y se visualiza de manera individual, el cuadro capturado se muestra en la figura 3.17.

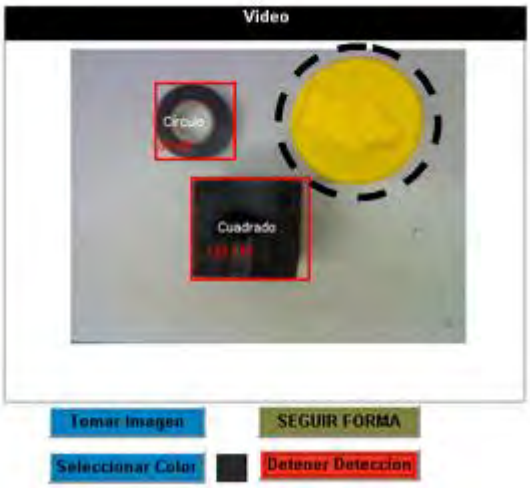


Figura 3. 17 Cuadro guardado en el WorkSpace

Después de la binarizacion con el color elegido se genera una matriz en la cual únicamente los pixeles correspondientes al color seleccionado son cambiados a “1”, y todos los demás se ponen en “0”. Se puede observar el resultado obtenido en la figura 3.18 donde se puede notar que la forma circular punteada es ignorada de debido a su color distinto a pesar de que es una figura que el programa puede reconocer.

Se puede visualizar que la cinta circular y el cuadro son del color muy aproximado al negro, ahora son blancos, a su vez esto mismo precede al siguiente paso que son las operaciones morfológicas.

Figura 3. 18 Resultado de la binarizacion únicamente de los objetos del color seleccionado

Las operaciones procedentes al paso de la figura 3.18 se ejecutan de la misma manera que en el proceso no automatizado (rellenado de agujeros, erosión, dilatación) la única diferencia es que este mismo proceso se repite indefinidamente, contando con la posibilidad de mover al objeto de su posición y seguir siendo detectado. Un cuadro capturado de la misma manera que el de la figura 3.18 se muestra en la imagen 3.19:

Figura 3. 19 Imagen después de las operaciones morfológicas

3.6.2.2 Envío de la pieza detectada

Posterior al proceso efectuado en la figura 3.19 se exporta dicha imagen a la función para el reconocimiento de forma, el cual aplica el mismo algoritmo de los apartados 3.3 y 3.4. Después de estos dos pasos ahora la comunicación con el modulo de E/S del manipulador es constante y automática, en otras palabras esto quiere decir que para que al programa de reconocimiento se le pueda controlar el tiempo de envió de la señal respectiva a la figura geométrica que ha sido detectada se utiliza la instrucción "FramesAcquired" la cual nos puede proporcionar cuantos cuadros han sido capturados y procesados hasta el momento, por lógica, entre mayor sea el parámetro de cuadros establecido para retardar el envío, será mayor el tiempo en que se envié la señal correspondiente a la figura detectada y viceversa .

Configurando un parámetro para que se procese 1 cuadro de imagen de cada 5 tomados por la cámara, se genera la siguiente tabla:

CUADROS	TIEMPO APROXIMADO(SEG)
40	13
31	10
25	8
18	6
15	5
10	3
5	1.5

Tabla 3. 1 Tiempos de procesamiento para sincronizar el manipulador –envío de figura

Como dato de interés, cabe mencionar que dividiendo el total de cuadros entre en tiempo de envío, se llega a que el programa procesa 3 cuadros/segundo en promedio aproximadamente. Tomando en cuenta este dato se puede programar en que instante del procesamiento se puede enviar la señal de la figura que se detecta debido a que esta

detección podría cambiar en cualquier momento; con todo esto se llega a la conclusión de que el momento más exacto para enviar la señal de la figura detectada es cuando el manipulador se posiciona justo por encima de la figura, evitando perder tiempo en la espera por una mala sincronización entre los tiempo en que el manipulador recibe la señal que lo enviara a recoger la configura detectada y el regreso a la nueva detección.

Sincronización Manipulador-Programa de reconocimiento

Al cronometrar el tiempo de las trayectorias que tiene que seguir el manipulador desde que reconoce al objeto, lo recoge, lo ordena en el lugar correspondiente y regresa a la siguiente detección (ciclo completo), se obtuvo que a máxima velocidad (2 m/s) el tiempo fue de 10 segundos (como se muestra en la figura 3.20), obteniendo variaciones solo en decimas, se concluyo que debido a esas ligeras diferencias de tiempo, la mejor opción para sincronizar entre un envió y otro es programar el parámetro de retardo a 5 cuadros, estos es, enviar la señal correspondiente para cada tipo de figura geométrica cada 1.5 segundos sin importar la posición. Debido a que el manipulador solo puede estar receptivo a este tipo de señal en un momento especifico durante el programa, se logra que si llegara a detectar una figura a mitad de camino, simplemente la ignora.

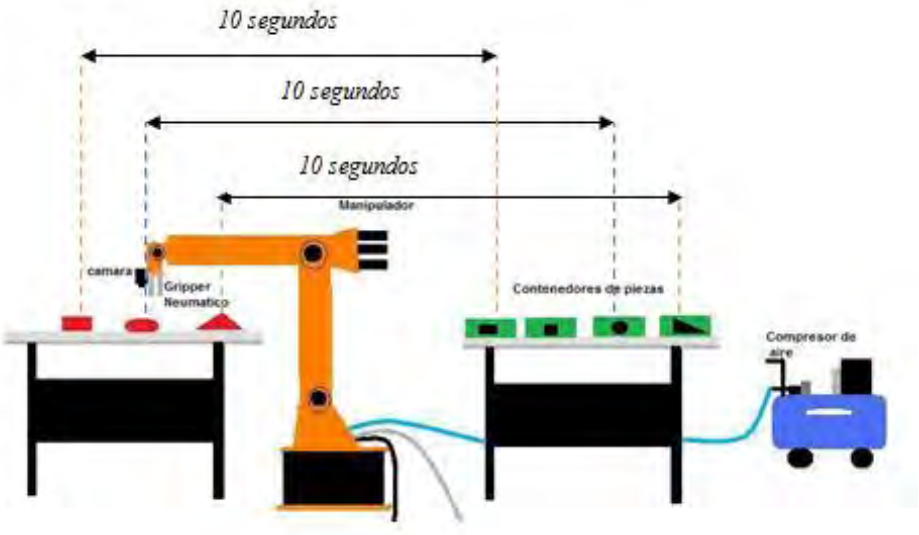


Figura 3. 20 Tiempo de trayectorias para cada figura

3.7 Conclusiones del capitulo

En el presente capitulo se desarrollaron 2 algoritmos de reconocimiento de forma, en el primero (el no automático), había la necesidad de que un operador ejecutara cada paso de dicho procedimiento, desde el preporcesamiento de la imagen hasta el reconocimiento y envío de pieza, lo que lo hacía un sistema funcional pero no totalmente automático por lo que se estableció la meta de poder hacerlo sin la necesidad de algún operador, lo cual se pudo realizar después de analizar varios programas de procesamiento de imágenes en tiempo real realizados en la maestría, lo que dio lugar a algoritmo de reconocimiento totalmente automático, desde el preprocesamiento de la imagen hasta el envío de señales, con lo que solo se necesita un operador para encender o apagar todo el sistema.

CAPITULO 4
Sistema Robótico KUKA KR-16

4.1Introducción

En el presente capitulo se describirá la última de las tres partes más importantes de este proyecto, el cual consiste en un sistema robótico KUKA KR16, que a su vez está formado por la unidad de control la cual es la encargada de recibir la señales de el programa de reconocimiento, el KCP (KUKA Control Panel) que sirve como interfaz con el usuario para las 2 formas de programación utilizadas (gestual y textual) además de proporcionar un medio para el manejo nivel software de las señales de entrada y salida, la otra parte consiste en el manipulador el cual es controlado por las dos anteriores y es el que tiene contacto físico directo con las piezas además de que se encarga de ejecutar las trayectorias para la correcta sujeción y acomodo de piezas .

Es importante señalar también que la parte correspondiente a la programación para la recepción de señales se realiza de una manera en que el manipulador solo este receptivo a ellas en un lapso corto en el que se ejecuta toda la lógica de manejo de trayectorias, lo que lo hace un sistema capaz de ignorar señales durante la ejecución de las mismas.

4.2 Sistemas de coordenadas

Durante el desplazamiento del robot los movimientos se realizan con respecto a un sistema de coordenadas.

Sistema de coordenadas especifico en los ejes

En el sistema de coordenadas específico de los ejes que se muestra en la figura 4.1, se puede desplazar individualmente cada uno de los ejes del robot en la dirección axial positiva o negativa. Este tipo de sistema se usa generalmente para movimientos largos y no tan exactos.

Figura 4.1 En este sistema los desplazamientos de cada eje son independientes

Sistema de coordenadas universal (WORLD)

El sistema de coordenadas WORLD (coordenadas universales) es un sistema de coordenadas cartesiano situado en un punto fijo. Funciona para movimientos más exactos como la calibración del *gripper* a diferentes sistemas. Además de servir como sistema de coordenadas de origen para los sistemas de coordenadas BASE y ROBROOT el cual se define más adelante.

Figura 4.2 En este sistema los movimientos son más finos y exactos que el sistema anterior

Sistema de coordenadas BASE

El sistema de coordenadas BASE es un sistema de coordenadas cartesiano. Este sistema se mide de tal modo que su origen se encuentra en una pieza o en un dispositivo.

Sistema de coordenadas TOOL

El sistema de coordenadas TOOL (figura 4.3) es un sistema de coordenadas cartesiano. Este sistema se mide de tal manera que su origen está situado en la herramienta que está usando el robot.

Figura 4.3 En este sistema las coordenadas se mueven respecto a un punto fijo en la herramienta

Sistema de coordenadas ROBROOT

El origen del sistema de coordenadas ROBROOT está siempre en la base del robot y está referido al sistema de coordenadas WORLD. Ello permite definir un corrimiento del robot hacia el sistema de coordenadas WORLD.

4.3 Tipos de movimientos

4.3.1 Movimiento punto a punto (PTP)

El posicionamiento del sistema de robot se produce aquí por el recorrido más corto entre dos puntos. Debido a que el movimiento en todos los ejes comienza y finaliza al mismo tiempo, los ejes deben ser sincronizados. Por este motivo, la trayectoria del robot no puede predecirse con exactitud.

Movimiento PTP con parada exacta

En los movimientos PTP con parada exacta, el robot se desplaza con exactitud hasta cada punto de destino como se observa en la figura 4.4.

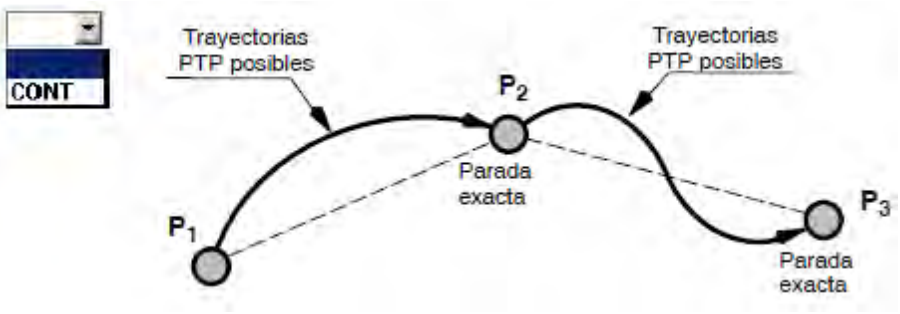


Figura 4.4 Movimiento PTP parada exacta

Movimiento PTP con posicionamiento aproximado

En el movimiento con posicionamiento aproximado, la unidad de control supervisa una zona de aproximación alrededor del punto de destino. En la figura 4.5, este es el punto P2. Cuando el punto de referencia de la herramienta se introduce en esta zona de aproximación, el movimiento del robot pasa al punto de destino de la instrucción de movimiento siguiente.

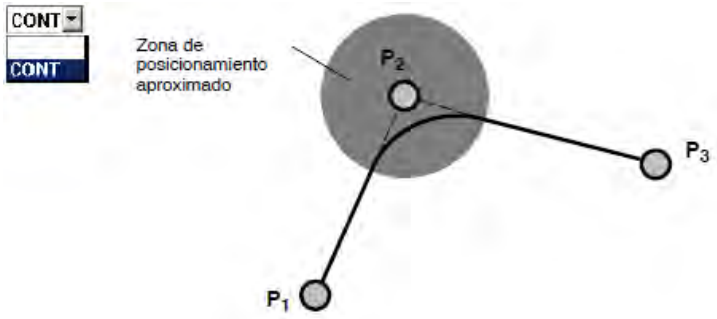


Figura 4.5 Movimiento PTP sin llegar a tocar el punto intermedio programado

4.3.2 Movimientos lineales (LIN)

En un movimiento lineal los ejes del robot se coordinan entre sí, de tal manera que el punto de referencia de la herramienta o de la pieza, se mueva a lo largo de una recta hacia el punto de destino. Los movimientos lineales se utilizan cuando se requiere para la aproximación de un punto un guiado de recorrido exacto con velocidad predeterminada.

Movimientos LIN con parada exacta

En movimientos LIN con parada exacta, el robot se desplaza a cada uno de los puntos de destino con posicionamiento exacto.

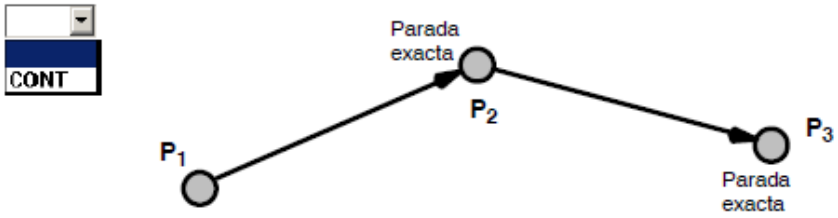


Figura 4.6 Movimiento LIN con parada exacta

Movimientos LIN con posicionamiento aproximado

En el movimiento con posicionamiento aproximado, la unidad de control supervisa una zona de aproximación alrededor del punto de destino. En el ejemplo de la figura 4.7, este es el punto P2. Cuando el punto de referencia de la herramienta se introduce en esta zona de aproximación, el movimiento del robot pasa al punto de destino de la instrucción de movimiento siguiente.

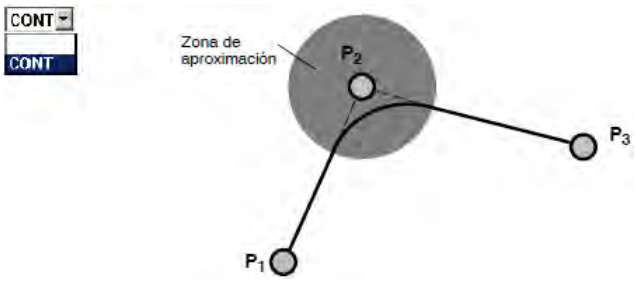


Figura 4.7 Movimiento LIN aproximado

4.4 Salidas y entradas Digitales en el KCP

Entradas

Antes de realizar la parte del programa en la que un ciclo infinito detecta la activación de algunas de las cuatro entradas utilizadas para esta automatización, se comprueba si las 4 señales (enviadas por la interfaz USB de +5V) de entrada de +24V son recibidas

correctamente por el modulo de E/S WAGO, esto se visualiza en el panel de control en la parte de indicación- E/S. Debido a que cuando estas entradas se activan son por un tiempo muy corto (en ms solo para esta aplicación) no se pudo visualizar en la imagen, pero viendo el KCP a simple vista se puede comprobar debido a que los círculos cambian a color rojo por un instante.

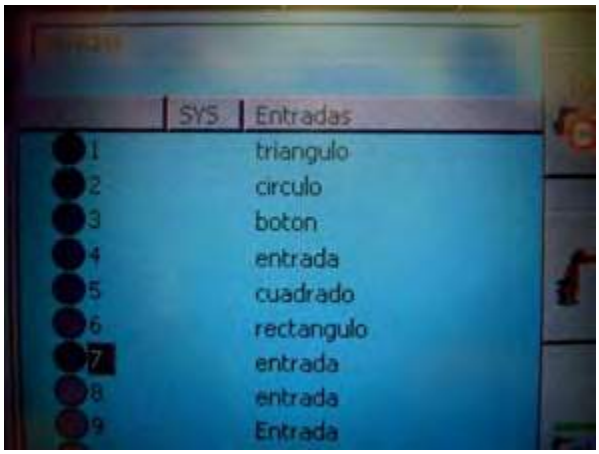


Figura 4.8 Las 4 entradas digitales correspondientes a cada figura

Después de la comprobación visual, se pueden utilizar dichas entradas digitales en el programa, siempre y cuando su valor sea manejado por una variable previamente declarada de tipo booleano para que de esta manera pueda ser manejado por los ciclos y las instrucciones condicionales.

Salidas

De igual manera que las entradas, las salidas son de +24v y se pueden visualizar en Indicación- E/S- salidas como se muestra en la figura 4.8. En esta foto debido a que la activación de cada una de ella se hace por un tiempo más prolongado que el de las entradas se puede visualizar más claramente, además son dados de alta los nombres para lo que son usadas cada una de ellas, con los cuales pueden también ser llamadas en el programa.

Figura 4.8 Las salidas 1 y 3 encargadas de manejar el Gripper Neumático y el sistema anticolisión

4.5 Gripper neumático y sujeción de piezas

Para poder acoplar las salidas de +24v que proporciona el modulo de E/S WAGO que van a la electroválvula la cual a su vez controla la apertura o cierre del gripper, se uso un relevador conectado directamente a la salida (Figura 4.9) y a su vez este activa a un eliminador el cual enciende o apaga la válvula neumática que es la encargada de dar dirección al aire comprimido para que el gripper tome la pieza o la suelte.

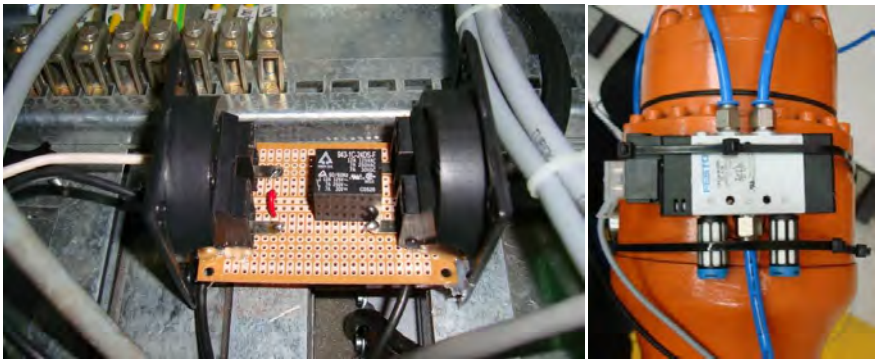


Figura 4.9 Relevador (izquierda) y electroválvula (derecha)

El diagrama de los componentes de la figura anterior se muestra en la imagen 4.10

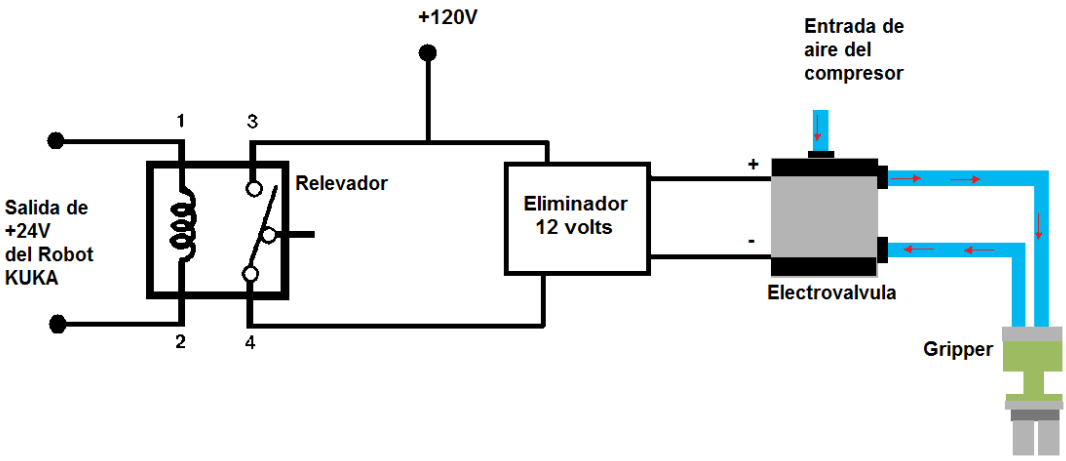


Figura 4.10 Diagrama electroneumático

Cuando el gripper se abre debido a que la electrovalvula cambio la direccion del aire comprimido, despues de que el programa de deteccion de forma determina el tipo de figura geometrica, el brazo robotico se orienta hacia la pieza mediante el tipo de coordenadas universales describiendo una linea recta mediante la instrucción LIN directamente hacia ella , para que despues se cierre el gripper justo en la coordenada para sujetar el objeto.

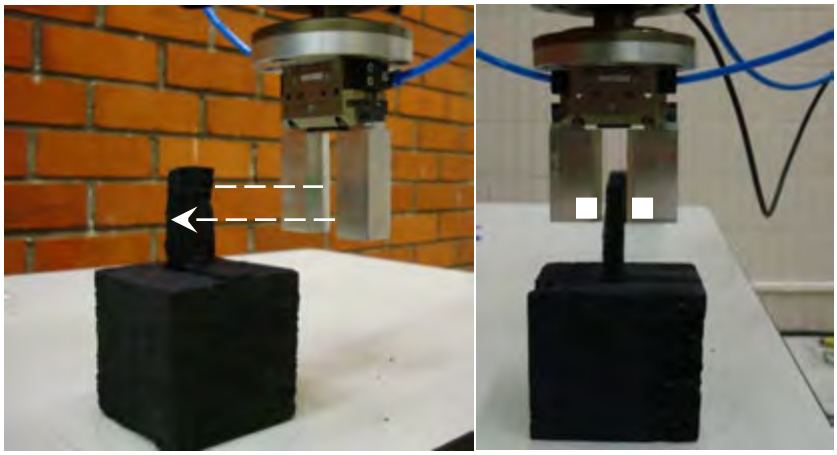


Figura 4.10 Alineación para sujetar la pieza

La pieza queda sujeta de la manera en que se muestra en la figura 4.11, pero cabe señalar que el tipo de sujecion puede cambiar dependiendo de la forma de los dedos del gripper.

Figura 4.11 Sujeción de la pieza con el gripper

4.6 Procedimiento para la selección

Para la selección de piezas, el manipulador puede ubicarse en tres posiciones diferentes, teniendo la posibilidad con esto de poder empacar tres piezas seguidas sin la necesidad de que la línea imaginaria sobre la cual se ubican las piezas vuelva a ser recargada con nuevos objetos, con esto no quiere decir forzosamente que deba de trabajar de esa misma manera, el programa está hecho para modificarse con una simple instrucción y así poder alterar su funcionamiento, de esta manera solo trabajara en una sola coordenada en especifico (suponiendo que una línea de producción solo lleva una fila única de piezas).

Figura 4.11 Ubicación del manipulador a la espera de la señal

Para evitar errores en la detección, el campo de visión que abarca la cámara debe ser lo bastante amplio para lograr que la pieza quede en el centro de la imagen, tomando en cuenta que ninguna parte de la imagen de la pieza quedar cortado en algún extremo al hacer la toma.

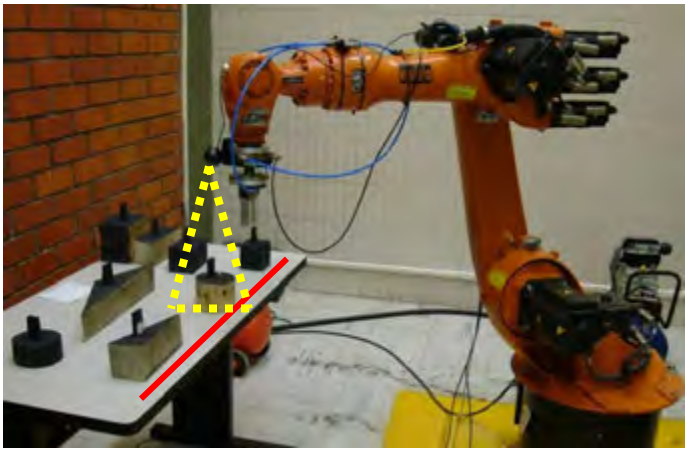


Figura 4.12 Línea imaginaria que forman las 3 piezas en turno

4.7 Ubicación y acomodo de piezas

Una vez que la pieza fue tomada por el gripper del manipulador, el paso siguiente es depositar la pieza en su lugar correspondiente y para esto sigue las trayectorias marcadas con color amarillo, donde se puede observar que 2 de ellas son líneas rectas (LIN) y la intermedia es una punto a punto (PTP), los primeros 2 movimientos se eligieron así debido a que se necesita mucho precisión al seguir el camino para ubicar la pieza, dado que no se tiene mucho margen de error en las coordenadas al tener una pieza del mismo tipo aproximadamente a 2 milímetros de distancia, reduciendo la posibilidad de posibles colisiones, pero las desventaja es que la velocidad máxima disminuye a 1.5m/s , el trayecto intermedio se eligió de esa manera debido a que el robot puede tomar máxima velocidad(2m/s) y no se necesita tanta precisión como al ordenar las figuras en las cajas.

4.7.1 Planificación de la trayectoria y control del movimiento del manipulador

Con el conocimiento de la cinemática y la dinámica de un manipulador con elementos series, es de gran importancia mover los actuadores de sus articulaciones para cumplir una

tarea deseada controlando al manipulador para que siga un camino previsto. Antes de mover el brazo, es de interés saber si hay algún obstáculo presente en la trayectoria que el robot tiene que atravesar (ligaduras de obstáculos) y si la mano del manipulador necesita viajar a lo largo de una trayectoria definida (ligaduras de trayectoria).

El problema del control de un manipulador se puede dividir convenientemente en dos subproblemas coherentes: el subproblema de planificación de movimiento (o trayectoria) y el subproblema de control del movimiento. La curva espacial que la mano del manipulador sigue desde una localización inicial (posición y orientación) hasta una final se llama la trayectoria o camino.

La planificación de la trayectoria (o planificador de trayectoria) interpola y/o aproxima la trayectoria deseada por una clase de funciones polinomiales y genera una secuencia de puntos de consignas de control en función del tiempo para el control del manipulador desde la posición inicial hasta el destino.

En general, el problema de control de movimientos consiste en: 1) obtener los modelos dinámicos del manipulador, 2) utilizar estos modelos para determinar leyes o estrategias de control para conseguir la respuesta y el funcionamiento del sistema deseado. Desde el punto de vista de análisis de control, el movimiento del brazo de un robot se suele realizar en dos fases de control distintas. La primera es el control del movimiento de aproximación en el cual el brazo se mueve desde una posición/orientación inicial hasta la vecindad de la posición/orientación del destino deseado a lo largo de una trayectoria planificada. El segundo es el control del movimiento fino en el cual el efector final del brazo interacciona dinámicamente con el objeto utilizando información obtenida a través de la realimentación sensorial para completar la tarea.

Los enfoques industriales actuales para controlar el brazo del robot tratan cada articulación como un servomecanismo de articulación simple. Este planteamiento modela la dinámica de un manipulador de forma inadecuada porque desprecia el movimiento y la configuración del mecanismo del brazo de forma global. Estos cambios en los parámetros del sistema controlado algunas veces son bastante significativos para hacer ineficaces las

estrategias de control por realimentación convencionales. El resultado de ello es una velocidad de respuesta y un amortiguamiento del servo reducido, limitando así la precisión y velocidad del efector final y haciéndolo apropiado solamente para limitadas tareas de precisión. Los manipuladores controlados de esta forma se mueven a velocidades lentas con vibraciones innecesarias. Cualquier ganancia significativa en el rendimiento de esta y otras áreas de control del brazo del robot requieren la consideración de modelos dinámicos más eficientes, enfoques de control sofisticados y el uso de arquitecturas de ordenadores dedicadas y técnicas de procesamiento en paralelo.

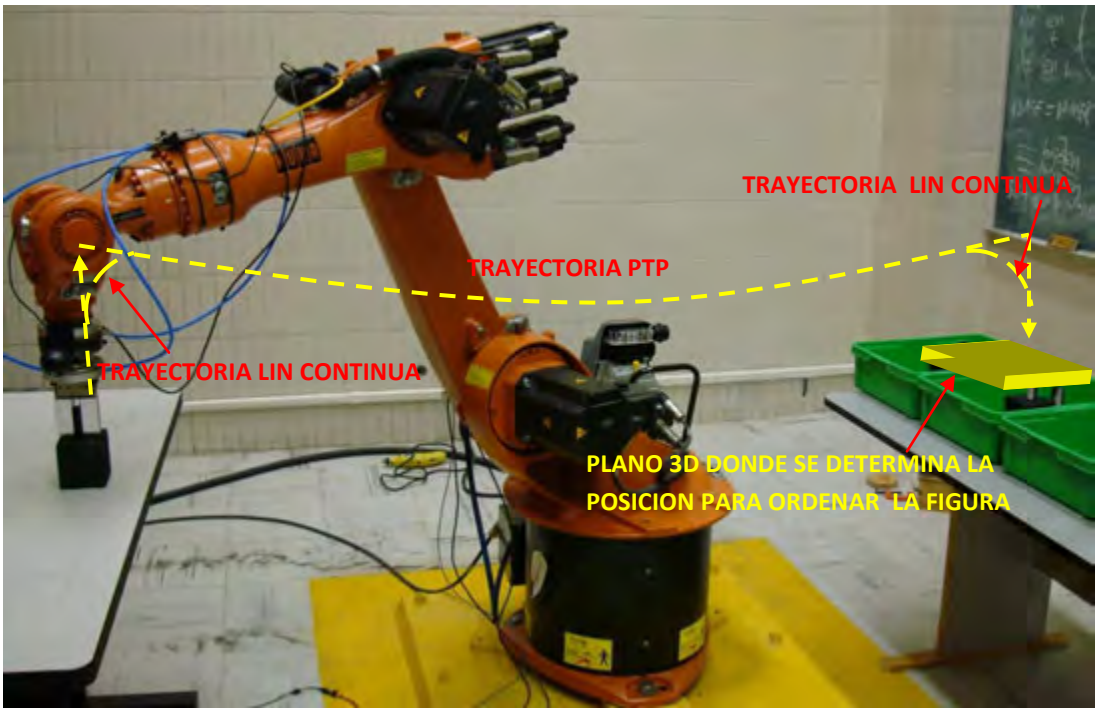


Figura 4.13 Camino recorrido para llegar a los contenedores

Después de ejecutar las 3 trayectorias representadas en la figura 4.13, el manipulador se posiciona en un plano 3D imaginario, en el cual tomara la decisión (tomando en cuenta que pieza ya ha sido depositada) de cuál será la coordenada en la que se ubicará la siguiente pieza.

Para ordenar las piezas cuadradas se eligio un acomodo en linea recta el cual se muestra con la linea punteada correspondiente a la figura 4.14 al centro de la caja, se eligio dicho orden de izquierda a derecha, pero pudo haber sido de derecha a izquiera, o de arriva abajo, teniendo en cuenta tambien que la pieza podria ser rotada +70º o -70º en los ejes X,Y

o Z para optimizar espacios ; lo cual tambien depende de la cantidad de piezas, forma y tamaño.

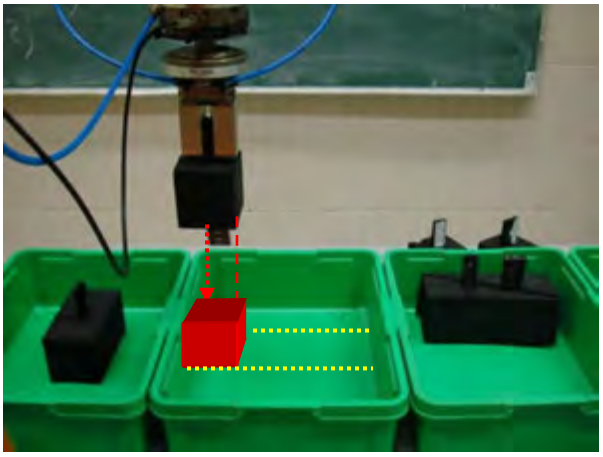


Figura 4.14 Orden de la primera pieza cuadrada

Para ordenar la segunda pieza (al centro) únicamente se recorre la coordenada aproximadamente 8 cm hacia la derecha con respecto a la posición anterior de la primera pieza, apuntando directamente en línea recta hacia la coordenada donde se va a ubicar dicha pieza, como se puede observar en la figura 4.15.

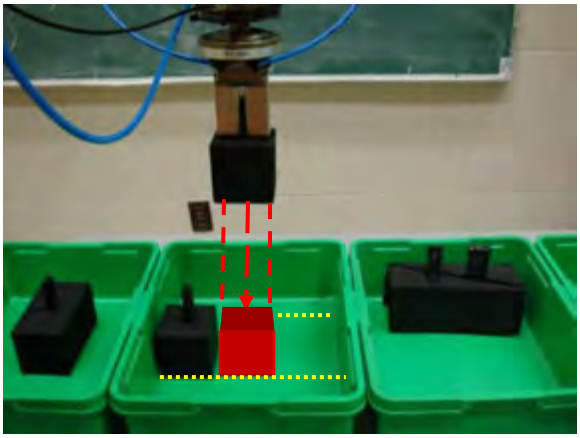


Figura 4.15 Acomodo de la segunda pieza cuadrada

La última pieza de la línea de 3 se muestra en la figura 4.16, de la misma manera que la anterior, el desplazamiento es de 8cm con respecto a la coordenada de la pieza intermedia para terminar un una instrucción LIN a una velocidad de 1m/s.

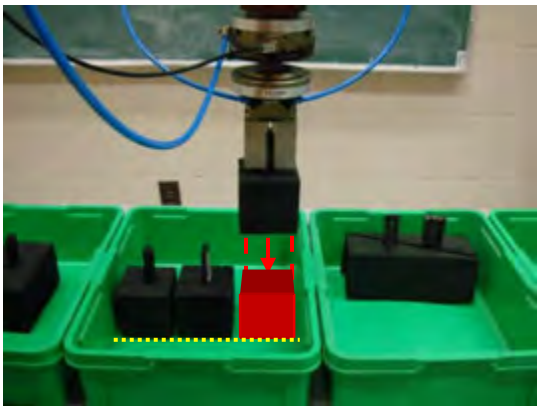


Figura 4.16 Acomodo de la tercera pieza

Para un acoplamiento distinto de las piezas, se utilizaron las figuras triangulares la cuales tienen un corte diagonal, el cual puede facilitar el acomodo debido a que se pueden adaptar 2 triángulos para formar un rectángulo, el primer paso para dicho procedimiento se puede observar en la figura 4.17 la cual muestra la misma toma pero de diferentes ángulos.

Figura 4.17 Acomodo de la primera pieza triangular

Con un giro de aproximadamente 100º con respecto a la posición del triangulo en la figura 4.17 se llega a la coordenada previa antes de ejecutar la trayectoria en línea recta hacia su posición final.

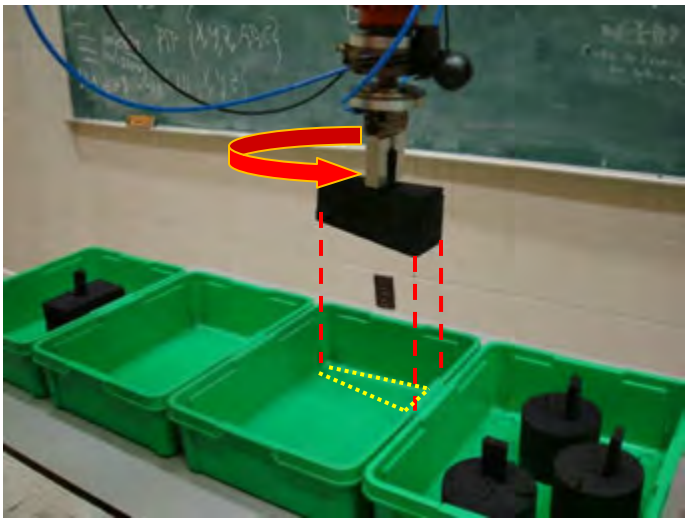


Figura 4.18 Acomodo de la primera pieza triangular

Al paso correspondiente a la figura 4.18 le sucede en paso de la figura 4.19 en el cual se ordenan las 2 restantes figuras triangulares, la primera de la izquierda completa la figura del rectángulo con una separación entre ellas de 2 mm, en la imagen de la derecha se vuelve a ordenar el triangulo a la orilla de la caja pero se podría seguir el mimo patrón de orden se así se quisiera.

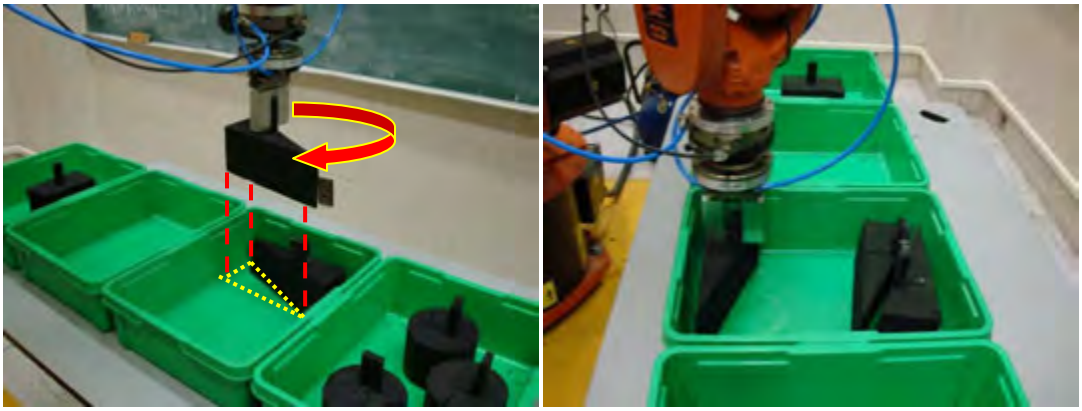


Figura 4.19 Distintas posiciones para piezas triangulares

4.8 Programación del Robot

El programa encargado de llevar el control del gripper así como todos los movimientos, posiciones, trayectorias, recepción de señales desde el programa de

reconocimiento y toda la lógica, es el algoritmo hecho en el KUKA System Software(KSS) Releáse 5.2, el cual consta de 350 líneas de código.

La lógica del programa consiste en que primero son inicializadas todas las variables de los diferentes tipos (Booleano, Entera, Axis), para que posteriormente se pase al primer ciclo infinito como se observa en la figura 4.20, con la instrucción Loop el cual controla que la totalidad del programa se esté repitiendo indefinidamente, el segundo ciclo infinito anidado dentro del primero genera que el manipulador valla a su primer coordenada para la detección y espere hasta que llegue alguna señal de cual figura geométrica está siendo detectada en tiempo real, en el momento en que pase dicha detección este primer ciclo se rompe debido a que se detecta que una entrada digital ha sido activada, consecuentemente este es el momento en que el manipulador llama a la función para enviar a la pieza a su lugar correspondiente y llegar al plano donde se decide el lugar exacto donde la pieza será ubicada. Posterior a que termina dicho ciclo en momento en que es llamada la función para enviar pieza, automáticamente el manipulador ya sabe cuál será su siguiente movimiento, el cual corresponde a la siguiente coordenada de detección para la segunda pieza, repitiendo todos los ciclos anteriores.

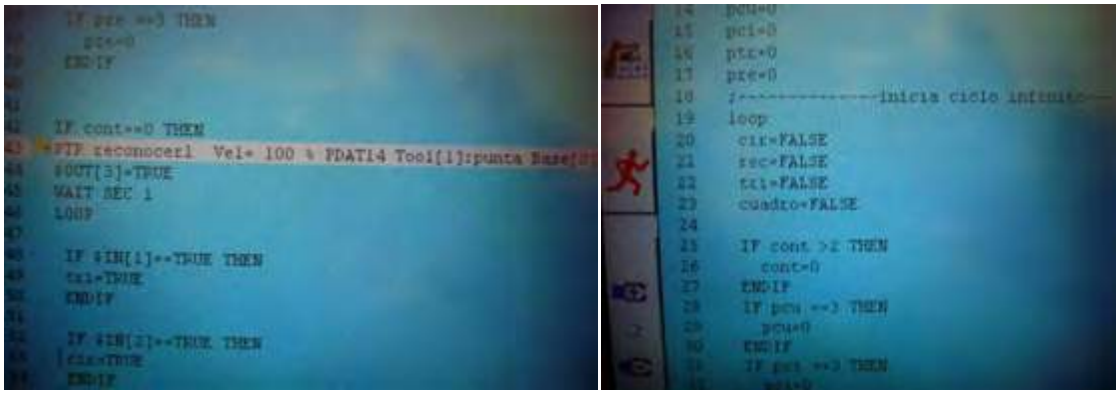


Figura 4.20 Inicio del programa (´Derecha) y primer detección (Izquierda)

En la figura 4.21 se muestra el diagrama de flujo el cual muestra 1/3 parte del programa, pero es suficiente para poder entender su funcionamiento debido a que las otras

2/3 partes utilizan la misma lógica lo único que cambia son las coordenadas para cada reconocimiento.

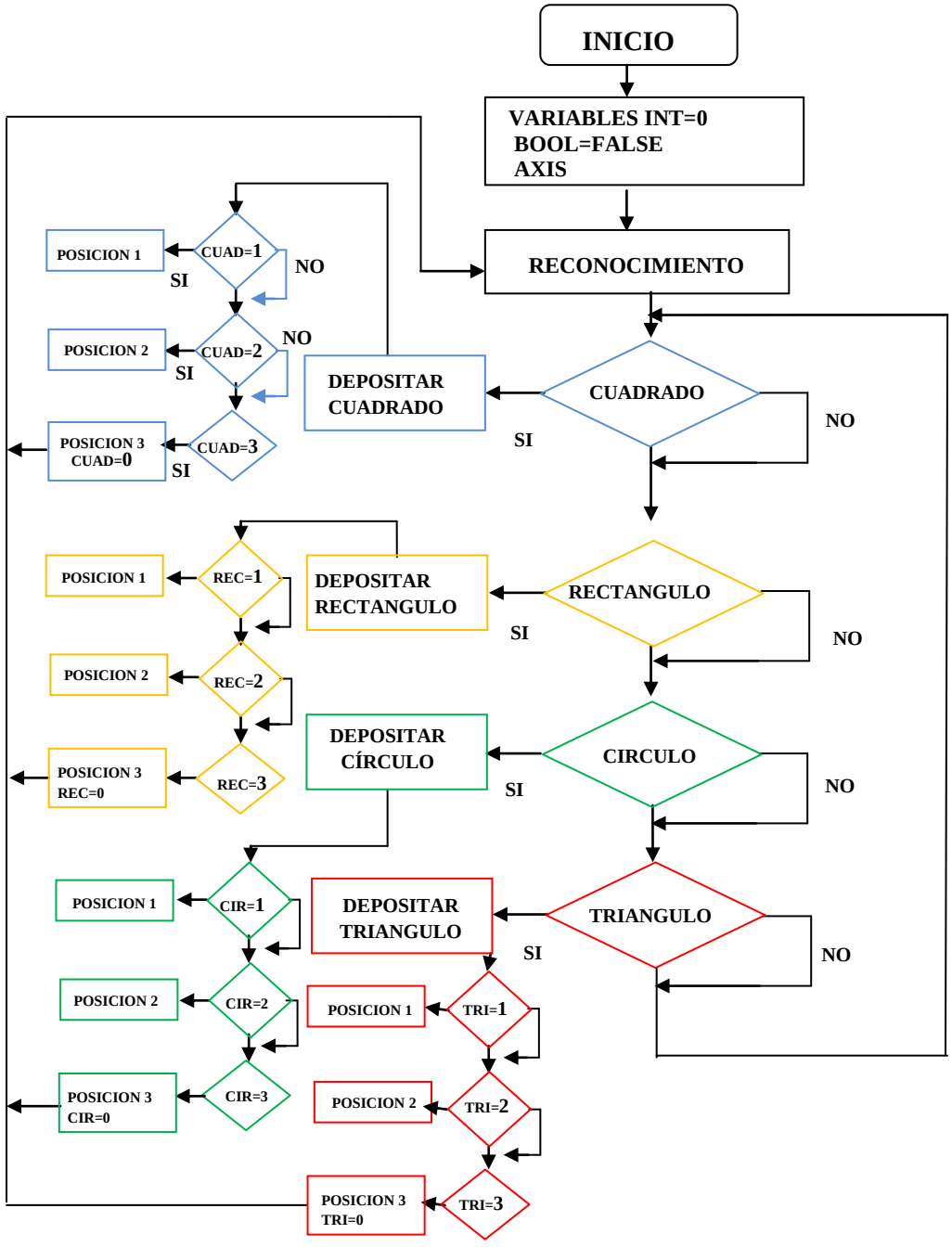


Figura 4.21 Diagrama de flujo para la detección

4.9 Sistema completo

En la imagen 4.22 se muestran todos los componentes del sistema robótico para la selección y manipulación de objetos a partir de su forma geométrica, donde se muestra el compresor que proporciona la presión de aire necesaria para el funcionamiento del Gripper, contenedores de piezas, brazo robótico, el sistema de visión artificial, ejemplos de piezas, interfaz USB, KUKA KCP, gabinete (unidad de control), PC donde se realiza el reconocimiento de todas las formas, finalizando con las conductos neumáticos que salen del compresor, el cable para comunicación KUKA KR16 - PC y por último el cable para la conexión modulo de control-KUKA KR16.

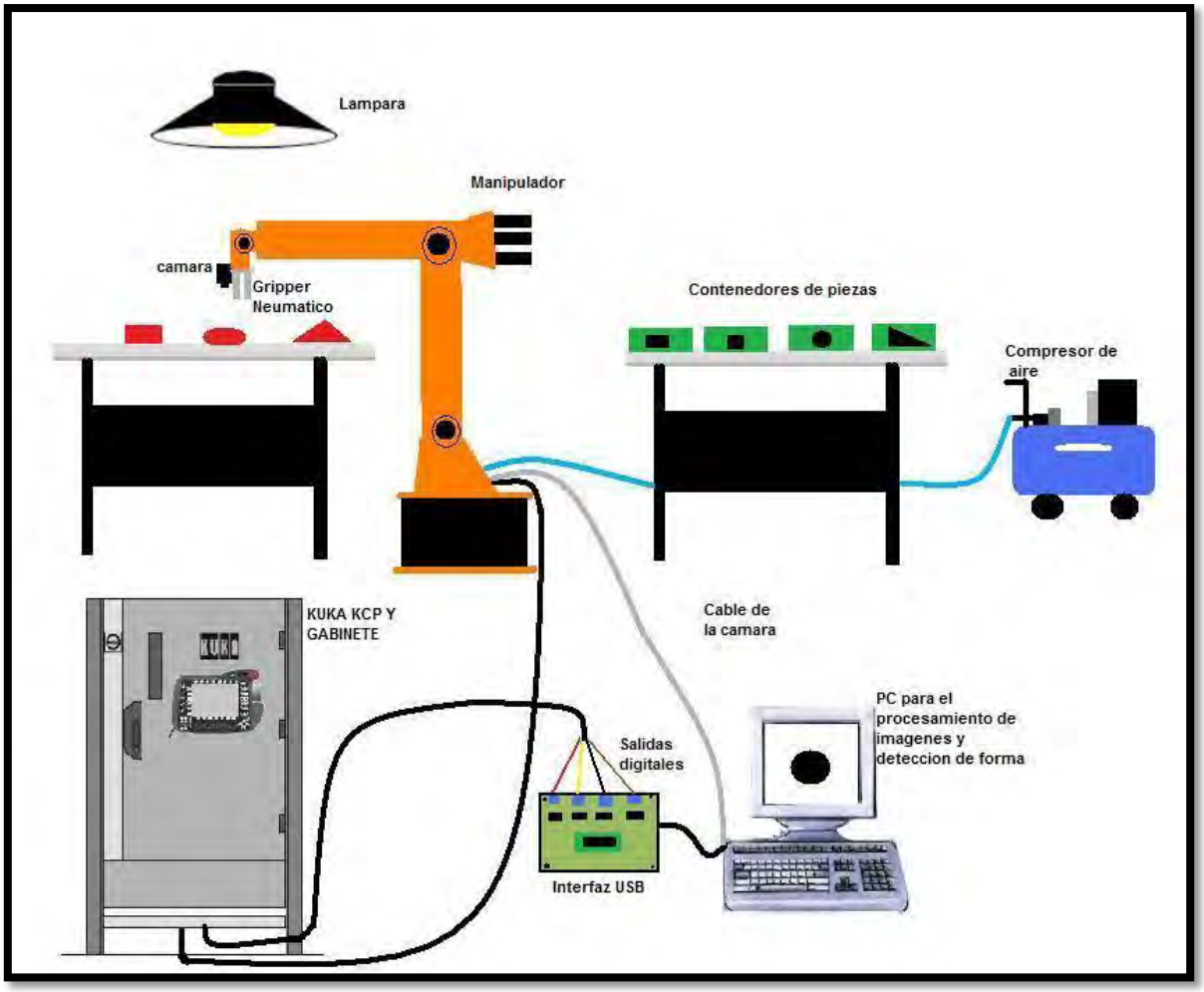


Figura 4.22 Sistema completo para la selección y manipulación

4.10 CONCLUSIONES DEL CAPITULO

Al terminar de escribir este capítulo se logro el objetivo de ensamblar y acoplar la última de las 3 partes importantes de esta automatización, lo que dio por resultado un sistema completo el cual es capaz de seleccionar piezas regulares e irregulares y manipular objetos regulares debido al tipo de gripper que se tiene disponible. Se realizó un algoritmo muy eficiente de 500 líneas aproximadamente, encargado del manejo de la lógica para las trayectorias y sujeción de piezas. También se implemento el circuito neumático encargado del controlar el funcionamiento del *Gipper* (sujetador) por medio de la electroválvula FESTO.

CAPITULO 5
CONCLUSIONES

En este desarrollo tecnológico multidisciplinario (**Sistema Optomecatronico**) se abarcaron varias áreas importantes como lo fue la óptica, electrónica, mecánica, computación, control, neumática, matemáticas, por mencionar las más importantes. Conjuntando todo esto se llego a realizar una automatización que tiene un impacto a nivel industrial, educativo y social debido a lo que su gran gama de aplicaciones le permite, pero sin dejar de lado que lo que ha sido desarrollado hasta ahorita puede seguir creciendo en cuanto a habilidades propias del proyecto así como en sus áreas de utilidad en el que puede ser aplicado.

Debido a que todas las partes excepto el manipulador fueron desarrolladas para este proyecto (Programa de reconocimiento, acoplamientos, modulo de comunicación, y programa del robot) en el laboratorio de robótica y electrónica se sabe que cada una de ellas funcionan en conjunto y sincronía para lograr un objetivo en común, pero esto no quiere decir que cada una de ellas no pueda funcionar de manera independiente para conjuntarse con otros proyectos de desarrollo tecnológico, dejando la posibilidad abierta de seguir encontrando aun mas aplicaciones que lleguen a resolver problemas cotidianos en los cuales la intervención del ser humano pudiera o no llevar muchas horas de trabajo, o utilizarlo en áreas donde se necesite mejorar la calidad de vida o comodidad de las personas, teniendo en cuenta también que en la división de educación se requiere en gran medida de este tipo de proyectos para que puedan servir como impulso para generar nuevas ideas que vallan contribuyendo a corto plazo a la automatización de cosas que antes parecían impensables, y a largo plazo a disminuir paulatinamente el atraso tecnológico del país.

Para ser mas especifico se puede concluir que para la parte de la interfaz de comunicación se logro algo que al principio parecía complicado, que era lograr establecer una sincronización estable entre el programa de reconocimiento y el manipulador, lo que permitió tener un sistema optomecatronico que pudiera unir varias disciplinas en un mismo proyecto, el cual es el objetivo de esta maestría, como consecuencia el manipulador puede

recibir, aceptar y entender las señales codificadas correspondientes a cada figura detectada, como se menciono en el capítulo 2, lo cual dio lugar a que las diferentes trayectorias del robot pudieran ser manejadas por el programa de reconocimiento, a causa de esto, se logra establecer uno de los objetivos más importantes de este proyecto, que consiste en que el manipulador sea manejado por software y no por un ser humano .

En el capítulo 3 se tenía la visión inicial de desarrollar un solo algoritmo, pero se fue observando que había la necesidad de otro para completar la automatización de manera más eficiente por lo que al final se conjuntaron para un objetivo común, dos algoritmos de reconocimiento de forma, en el primero (el no automático), había la necesidad de que un operador ejecutara cada paso de dicho procedimiento, desde el preprocesamiento de la imagen hasta el reconocimiento y envío de pieza, lo que lo hacía un sistema funcional pero no totalmente automático por lo que se estableció la meta de poder hacerlo sin la necesidad de algún operador, lo cual se pudo realizar después de analizar varios programas de procesamiento de imágenes en tiempo real realizados en la maestría, lo que dio lugar a algoritmo de reconocimiento totalmente automático, desde el preprocesamiento de la imagen hasta el envío de señales, con lo que solo se necesita un operador para encender o apagar todo el sistema.

En el capítulo 4 correspondiente al control del manipulador y recepción de señales fue necesario también desarrollar 2 programas iniciales y al final uno que conjuntaba a los dos anteriores para concluir con un tercer programa más complejo pero mucho más eficiente y capaz de ejecutar el mando de las tareas de una manera que quedara abierto para poder funcionar con algún otra señal digital externa, al final de el desarrollo de esta parte importante se logro el objetivo de ensamblar y acoplar esta parte importante de la automatización, lo que dio por resultado un sistema completamente automático capaz de seleccionar piezas regulares e irregulares y manipular objetos regulares solamente debido al tipo de gripper que se tiene disponible. Se realizó un algoritmo muy eficiente de 500 líneas aproximadamente, encargado del manejo de la lógica para las trayectorias y sujeción de piezas. Es importante señalar también que se realizo la instalación del circuito neumático encargado del controlar el funcionamiento del Gipper (sujetador) por medio de la electroválvula FESTO.

5.1 Aplicaciones Extras

Las características más importantes a tomar en cuenta a la hora de usar un robot manipulador para una tarea en específico son las siguientes:

Resolución: Mínimo incremento que puede aceptar la unidad de control del robot que para el caso del KUKA KR16 es de 0.005º en movimientos axiales o .1 mm para movimientos en el plano cartesiano. Su valor está limitado por la resolución de los captadores de posición y convertidores A/D y D/A, por el número de bits con los que se realizan las operaciones aritméticas en la CPU y por los elementos motrices, si son discretos.

Precisión: Distancia entre el punto programado y el valor medio de los puntos realmente alcanzados al repetir el movimiento varias veces con carga y temperatura nominales que para el caso del KUKA en caso de llegar a disminuir puede ser calibrado de nuevo. Si esto no se realiza esto con constancia podrían originar errores en la precisión o en la sincronización por poner un ejemplo, también dicha precisión puede verse afectada por deformaciones de origen térmico y dinámico, errores de redondeo en el cálculo de la transformación cinemática, errores entre las dimensiones reales y teóricas del robot, etc.

Repetibilidad: Radio de la esfera que abarca los puntos alcanzados por el robot tras suficientes movimientos, al ordenarle ir al mismo punto de destino programado, con condiciones de carga y temperatura iguales. El error de repetibilidad es debido fundamentalmente a problemas en el sistema mecánico de transmisión como rozamientos, histéresis, zonas muertas (*backlash*).

Otro punto importante a tomar en cuenta es el **área de trabajo** que abarca del robot debido a que para ciertas aplicaciones no se necesita llegar a algunos puntos en específico y podría invertirse en un robot de largo alcance, el cual se estaría desperdiciando en rendimiento y dinero, a su vez también se podría adquirir un robot de corto alcance, siendo insuficiente para las tareas requeridas.

Los **grados de libertad** son otro punto a tomar en cuenta debido a que pueden llegar a estorbar para ciertas aplicaciones o a hacer falta para mayor movilidad para otras. Comúnmente se usan entre 3 (paletizado) y 6 (pintura, soldadura de arco) y el coste es directamente proporcional al número de grados de libertad.

Una primera aplicación en la cual se puede utilizar este sistema manipulador y seleccionador de objetos es en la detección de piezas armadas circulares que no estén completas, calibrando el sistema para el color y forma de una pieza que se encuentre en el centro, como se puede observar en la figura 4.23 y 4.24.

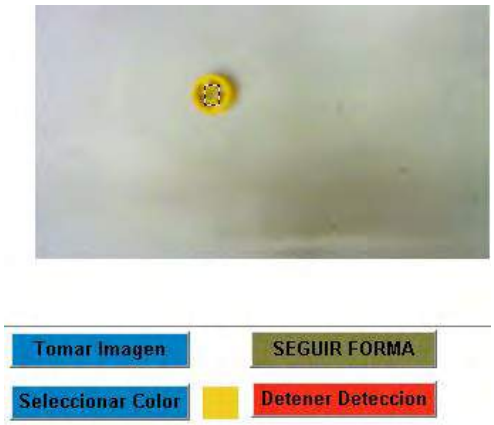


Figura 4.23 Calibración del sistema para el color amarillo como una pieza central

Si se usa un Gripper distinto al original para esta aplicación (Por ejemplo un gripper de 3 dedos), se puede manipular piezas más pequeñas y frágiles, el programa de detección seguiría funcionando de la misma forma, descartando piezas mal ensambladas.

Figura 4.24 Llanta de juguete armada correctamente a la izquierda

Manipulación de figuras irregulares

Otra aplicación muy interesante en la que se puede usar este proyecto es en la manipulación de formas irregulares como la de la figuras 4.24 por poner un ejemplo, esto es debido a que el programa está diseñado para asignar siempre la figura regular que más se le parezca a la imagen segmentada que se analiza. Con esto se logra poder empacar y ordenar figuras que no tienen que ser necesariamente formas geométricas regulares perfectas, como se puede observar en la figura 4.25 donde un huevo de granja es descrito por una forma irregular, en la cual el programa puede detectarlo como un circulo debido a su proximidad en área.

Figura 4.24 Figura de un huevo con bajo contraste (izquierda) y alto contraste (derecha)

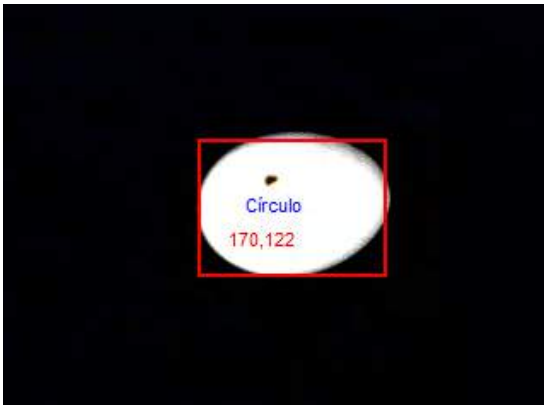


Figura 4.25 Asignación de la forma a la figura irregular

Descartar piezas sin tapa

Una aplicación interesante para la detección de forma muy parecida a la primera pero ahora utilizada en procesos industriales en los cual se puede tomar video de todas las botellas que deban de llevar tapa, asignando la formas de un cuadrado a dichas tapa , como se observa en la figura 4.26.

Figura 4.26 Pieza con tapa

Cuando se obtiene la referencia de que las botellas que están con tapa en su parte superior solo se detectan como un cuadro, entonces se puede pasar a seleccionar dichas piezas debido a que las botellas sin tapa denotaran una figura circular en su punta, como se observa en la figura 4.27 programando al manipulador para que las descarte tomando en cuenta que sería más efectivo usar un gripper de 3 dedos.

Figura 4.27 Pieza sin tapa y forma irregular en su punta

5.2 Resultados

Diseño e implementación de una nueva plataforma de visión

Esta parte incluye la selección de los componentes necesarios que son: el robot manipulador KUKA KR16, la cámara web digital Logitech, así como el diseño del entorno de selección de piezas regulares e irregulares.

Desarrollo de un nuevo algoritmo de reconocimiento

Este nuevo desarrollo encargado tanto del reconocimiento de forma como la comunicación con el manipulador, el cual aplica 2 técnicas diferentes de reconocimiento de forma basadas en determinar su área y medidas de los ejes.

Implementación del sistema electroneumático y acoplamientos

Se instalo un *gripper* neumático de 2 dedos paralelos y sus acoplamientos de salida desde el modulo WAGO, un sistema anticolisión, así como una válvula electroneumática de control para el *gripper* marca FESTO, terminando con mangueras y compresor de aire de 2HP.

Diseño del modulo de comunicación USB- KUKA KR16

Modulo realizado con el microcontrolador PIC18F2550 y 4 relevadores, encargado de recibir del puerto USB y codificar la información enviada desde el software para ser enviada al controlador del manipulador.

Diseño del programa del manipulador

Se diseño un programa encargado de captar señales digitales desde el modulo de comunicación USB de una manera que el manipulador las captara solo en determinado tiempo del transcurso del programa, ignorando dichas señales al estar ejecutando tanto la recolección como el acomodo de las piezas.

5.3 Trabajo futuro

Algunos proyectos a realizar para continuar trabajando en esta línea de investigación son:

1. Utilizar 2 cámaras digitales diferentes para controlar los movimientos del manipulador a partir de una imagen (controlador visual), ubicándolas una en la parte superior de la mesa de detección y otra en la parte superior de la mesa con los contenedores para acomodar las piezas.
2. Manipulación de diferentes tipos de piezas con un gripper neumático adaptable a la forma del objeto.
3. Desarrollar un sistema de agarre para que el manipulador aprenda por si solo a tomar las piezas de la forma correcta.
4. Desarrollar un sistema de bandas (instalación y sincronización) para cumplir la tarea de alimentar automáticamente las piezas a las diferentes posiciones de detección.
5. Implementar un sistema de adquisición y procesamiento de imágenes en tiempo real el cual sea lo suficientemente rápido para que la visualización de los resultados sean lo más cercano posible al tiempo real, disminuyendo desfasamiento de tiempo debido al procesamiento.
6. Sumarle a la actual herramienta (Gripper de 2 dedos paralelos) en uso, diferentes tipos de herramientas de sujeción, así como láseres, revolver y sujetadores de vacio para tomar varios objetos a la vez.

Referencias

[1] Jorge De Felice, Instituto Universitario de Robótica Física y Tecnología Experimental Albert Einstein, 2000.

[2] José Ma. Angulo, Robótica Práctica Tecnología y Aplicaciones, Ed. Paraninfo.

[3] R.C González, C.S.G. LEE, Robótica: Control, Detección, Visión e Inteligencia K.S. FU McGraw Hill.

[4] P. Castillo-García, Plataforma de control visual para servomecanismos, Tesis de maestría, CINVESTAV-IPN, 2000.

[5] Marco Antonio Moreno Armendáriz, Visión artificial estero con aplicación al control de un brazo de robot, Tesis de doctorado, CINVESTAV-IPN, 2003.

[6] Iluminación para aplicaciones de visión artificial Visión artificial, Universidad Nacional de Quilmes – Ing. en Automatización y Control Industrial, 2005.

[7] MICROCHIP-USB Firmware Users Guide, protocolo USB, ingeniería en microcontroladores.

[8] John Hyde Wiley, USB DESIGN BY EXAMPLE - A practical guide to building I/O devices ,2008.

[9]Atmel Corporation, USB CDC demonstration UART to USB Bridge, 2008.

[10]"The sampling and reconstruction of time-varying imagery with application in video systems" E.Dubois. Proceedings of the IEEE, vol.73, nº.4, April 1985, pp.502-522

[11] R.DeHoff , "The essential guide to digital video capture" .

[12] Practicas de Laboratorio de la Universidad La Salle, Talleres y Laboratorios, Laboratorio de Robótica. Universidad La Salle, 1999.

[13] Análisis morfológico en imágenes, Universidad Miguel Hernández.

[14] Jesús Cáceres Tello, "La visión artificial y las operaciones morfológicas en imágenes binarias", Universidad de Alcalá.

[15] Gonzalo Luzardo M, Segmentación de imágenes basado en etiquetación de pixeles.

[16] Grupo de tecnología industrial, División de Ingeniería de Sistemas y Automática, Reconocimiento de objetos, Universidad Miguel Hernández.

[17] David García Pérez, Descripción de regiones, Grupo de visión artificial.

[18] Grupo de tecnología industrial, División de Ingeniería de Sistemas y Automática, Segmentación de imágenes, Universidad Miguel Hernández.

[19] Eddie Angel Sobrado Malpartida, Sistema de visión artificial para el reconocimiento y manipulación de objetos utilizando un brazo robot, Tesis de maestría, Pontificia Universidad Católica del Perú, 2003.

[20] Grupo de tecnología industrial, División de Ingeniería de Sistemas y Automática, Sistemas de visión artificial, Universidad Miguel Hernández.

[21] Rawin Rojvanit, Migrating Applications to USB from RS-232 UART with Minimal Impact on PC software, Microchip Technology Inc.

Apéndice A

Programa en MatLab para el reconocimiento de forma.

```
function varargout = principal2(varargin)
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @principal2_OpeningFcn, ...
    'gui_OutputFcn',  @principal2_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

% ----------------- COMANDOS INICIALES DEL PROGRAMA--------------------.

```
function principal2_OpeningFcn(hObject, eventdata, handles, varargin)
% Centrar GUI
movegui(hObject,'center')
img=imread('KUKA.jpg');
axes(handles.axes3)
imshow(img)
axis off
img=imread('camara.jpg');
axes(handles.axes4)
imshow(img)
axis off

% Eliminar ejes de los axes
set(handles.axes1,'XTick',[],'YTick',[])
set(handles.axes2,'XTick',[],'YTick',[])
% Desactivar el botones
set(handles.inicio_b,'Enable','off')
% set(handles.tomar_b,'Enable','off')
set(handles.binarizar_b,'Enable','off')
set(handles.morfologia_b,'Enable','off')
set(handles.umbral_s,'Enable','off')
set(handles.segmenta_b,'Enable','off')
set(handles.reconocer_b,'Enable','off')
set(handles.enviar_b,'Enable','off')

% Choose default command line output for principal2
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

% --- Outputs from this function are returned to the command line.
function varargout = principal2_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
```

```
% ------- FUNCIÓN DEL BOTÓN DE SELECCIÓN DE CÁMARA----------------------.

function sel_camara_b_Callback(hObject, eventdata, handles)
% Abrir GUI para seleccionar la cámara a usar
sel_camara
% Esperar hasta
uiwait
% Activar el botón de inicio una vez selecionada la cámara
set(handles.inicio_b,'Enable','on')
% Traer las características de la cámara
global id es_web_ext
set(handles.text1,'String',id)

% --------------------- FUNCIÓN DEL BOTÓN INICIO -----------------------.

function inicio_b_Callback(hObject, eventdata, handles)
% Traer las características de la cámara
global id es_web_ext
% Crear objeto de video
handles.vidobj = videoinput('winvideo',id,'RGB24_320x240');
% Arrancar objeto de video
start(handles.vidobj);
% Actualizar estructura handles
guidata(hObject, handles);
% Obtener resolución
vidRes = get(handles.vidobj, 'VideoResolution');
% Obtener el número de bandas
nBands = get(handles.vidobj, 'NumberOfBands');
% Mostrar la imagen en el axes
hImage = image(zeros(vidRes(2), vidRes(1), nBands), 'Parent',...
    handles.axes1);
% Visualización del video
preview(handles.vidobj,hImage);
% Activar el botón de captura de imagen a procesar
set(handles.tomar_b,'Enable','on')
guidata(hObject, handles)

% ------------------------------FUNCIÓN DE CAPTURA DE IMAGEN------------.

function tomar_b_Callback(hObject, eventdata, handles)
try
    % Adquirir imagen de la webcam
    handles.rgb = getsnapshot(handles.vidobj);
catch
    % Si la cámara no está conectada, escoger una foto del disco rom.
    [no ru]=uigetfile({'*.tif';'*.jpg'},'Seleccionar imagen');
    handles.rgb =imread(fullfile(ru,no));
end
% Update handles structure
guidata(hObject, handles);
% Mostrar la imagen capturada
image(handles.rgb,'Parent',handles.axes2);
% Eliminar los axes
axes(handles.axes2)
axis off
% Mapa de colores gris
```

```
colormap(gray)
% Activar el botón para binarizar la imagen
set(handles.binarizar_b,'Enable','on')
set(handles.umbral_s,'Enable','on')


% ---------------------- FUNCIÓN DE BINARIZACIÓN----------------------.
function binarizar_b_Callback(hObject, eventdata, handles)
% Importar imagen procesada en la función anterior
img=handles.rgb;
% Tomar el valor del slider (umbral)
umb=get(handles.umbral_s,'Value');
% Binarizar la imagen con el umbral adquirido
handles.binaria=im2bw(img,umb);
%~im2bw(img,umb);
% Mostrar la imagen capturada
imagesc(handles.binaria,'Parent',handles.axes2);
axes(handles.axes2)
% Eliminar los axes
axis off
% Activar el botón para filtrar la imagen
set(handles.morfologia_b,'Enable','on')
colormap(gray)
guidata(hObject,handles)

% --------------- FUNCIÓN DEL SLIDER----------------------------------.
function umbral_s_Callback(hObject, eventdata, handles)
% Importar imagen
img=handles.rgb;
% Tomar el valor del slider
umb=get(handles.umbral_s,'Value');
% Binarizar en base al umbral
handles.binaria=~im2bw(img,umb);
% Mostrar la imagen capturada
imagesc(handles.binaria,'Parent',handles.axes2);
axes(handles.axes2)
% Eliminar los axes
axis off
colormap(gray)
% Update handles structure
guidata(hObject, handles);

% ----------- PROCESAMIENTO MORFOLÓGICO DE LA IMAGEN----------------.

function morfologia_b_Callback(hObject, eventdata, handles)
% Importar imagen binaria
bw=handles.binaria;
% Eliminar áreas menores a 100 pixeles
bw=bwareaopen(bw,100);
% Eliminar áreas pegadas al borde (objetos incompletos)
bw=imclearborder(bw);
% Operación morfológica para eliminar agujeros
bw  = bwmorph(bw,'open');
% Eliminar ruido sobre imagen
handles.bw  = bwmorph(bw,'close');
% Rellenar agujeros sobre las formas
```

```
handles.bw  = imfill(bw,'holes');
% Mostrar la imagen capturada
imagesc(handles.bw,'Parent',handles.axes2);
axes(handles.axes2)
% Eliminar los axes
axis off
colormap(gray)
set(handles.segmenta_b,'Enable','on')
% Update handles structure
guidata(hObject, handles);
```

---------- Segmentacion de Imagenes----------

```
function segmenta_b_Callback(hObject, eventdata, handles)
% Importar imagen  procesada en la función anterior
binaria=handles.bw;
% Etiquetar cada objeto y enumerarlos
[L Ne]=bwlabel(binaria);
% Obtener de cada objeto: área, centroide y caja
prop=regionprops(L);
% Enmarcar cada objeto y escribir el valor de su centro de masa
for n=1:Ne
    % Obtener el centro de masa
    centro=round(prop(n).Centroid);
    x=centro(1);
    y=centro(2);
    % Escribir la coordenada de cada objeto
    text(x-35,y,[num2str(x),',',num2str(y)],'FontSize',8,'Color','r')
    % Graficar el rectángulo (caja) sobre cada objeto

rectangle('Position',prop(n).BoundingBox,'EdgeColor','r','LineWidth',2)
end
set(handles.reconocer_b,'Enable','on')
guidata(hObject,handles)
```

---------- RUTINA DE RECONOCIMIENTO----------

```
function reconocer_b_Callback(hObject, eventdata, handles)
% Umbral de diferencia para distinguir entre un cuadrado y un rectángulo
umb=20;
% Importar imagen procesada
bina=handles.bw;
% Etiquetar cada objeto y enumerarlos
[L Ne]=bwlabel(bina);
% Obtener propiedades de cada objeto
prop=regionprops(L,'Image','MinorAxisLength','MajorAxisLength','Area','Ce
ntroid');
% Matrices vacías para que acumulen las coordenadas de cada objeto
cT=[ ];
cC=[ ];
cR=[ ];
cCu=[ ];
% Ciclo de identificación
for n=1:Ne
    % Tomar la imagen "n"
    img=prop(n).Image;
```

```
    % Redimensionar la imagen a 40x40 pixeles
    rsz=imresize(img,[40,40]);
    % Sumar el número de pixeles (área) de cada objeto
    area(n)=sum(sum(rsz));
    % Cada forma tendrá un área enmarcada dentro de ciertos valores
    if area(n)<1000 % Para el caso de triángulo
(base*altura/2=40*40/2=800)
        Cc=prop(n).Centroid;
        cT=[cT;Cc];
    elseif area(n)>1000&&area(n)<1285 % Para el caso del círculo
(pi*(rad)^2)
        Cc=prop(n).Centroid;
        cC=[cC;Cc];
    else % Para el caso de cuadrado y rectángulo
        Cc=prop(n).Centroid;
        largo=prop(n).MajorAxisLength; % Valor del eje mayor
        ancho=prop(n).MinorAxisLength;% Valor del eje menor
        % Diferencia entre los ejes precedentes
        dife=abs(largo-ancho);
        if dife<umb %Cuadrado
            cCu=[cCu;Cc];
        else %Rectángulo
            cR=[cR;Cc];
        end
    end
end
% Presentar la imagen binaria
imagesc(bina,'Parent',handles.axes2)
% Quitar los ejes
axes(handles.axes2); axis off
hold on
% plot(cT(:,1),cT(:,2),'r^')
% plot(cC(:,1),cC(:,2),'go')
% plot(cCu(:,1),cCu(:,2),'bs')
% plot(cR(:,1),cR(:,2),'mp')

%------------------ Graficar los nombres de cada figura------------------

if ~isempty(cT)
    text(cT(:,1)-30,cT(:,2),'Triang','FontSize',8)
end
if ~isempty(cC)
    text(cC(:,1)-40,cC(:,2),'Círculo','FontSize',8)
end
if ~isempty(cCu)
    text(cCu(:,1)-20,cCu(:,2),'Cuadrado','FontSize',8)
end
if ~isempty(cR)
    text(cR(:,1)-50,cR(:,2),'Rectangulo','FontSize',8)
end
hold off
%  Exportar las coordenadas de cada figura para enviarlas por el puerto
USB
%  (en modo CDC)
handles.c_tria    = cT;
handles.c_circ   = cC;
handles.c_cuad = cCu;
```

```
handles.c_rect   = cR;
% Mostrar el resultado en un cuadro de diálogo
%msgbox({['Circulos: ',num2str(size(cC,1))],['Rectangulos:
',num2str(size(cR,1))],...
   % ['Cuadrados: ',num2str(size(cCu,1))],  ['Triángulo:
',num2str(size(cT,1))]},...
   % 'Forma')
% Activar botón de envío
 set(handles.enviar_b,'Enable','on')
%     Actualizar estructura handles
guidata(hObject, handles)

% ----------------- COMUNICACIÓN SERIAL---------------------------------

function enviar_b_Callback(hObject, eventdata, handles)
% Importar los datos del proceso anterior
c_tria   =round(handles.c_tria);
c_circ  =round(handles.c_circ);
c_cuad =round(handles.c_cuad);
c_romb =round(handles.c_rect);

% 00 --- circulo
% 01 --- cuadrado
% 10 --- rectangulo
% 11 --- triangulo

if isempty(c_tria) % Verificar si hay datos para enviar
else
    for n=1:size(c_tria,1)
        % FWRITE escribe los datos en el puerto serial
        %fwrite(handles.SerPIC,num2str(c_tria(n,:)));
        % 11 --- triangulo
        %fwrite(handles.SerPIC,num2str(1));
        fwrite(handles.SerPIC,'a');
        % Pausa para que el robot
        pause(1)
    end
end
%  - - -
if isempty(c_circ) % Verificar si hay datos para enviar
else
    for n=1:size(c_circ,1)
        %fwrite(handles.SerPIC,num2str(c_circ(n,:)));
        % 00 --- circulo
        fwrite(handles.SerPIC,'b');
        pause(1)
    end
end
%  - - -
if isempty(c_cuad) % Verificar si hay datos para enviar
else
    for n=1:size(c_cuad,1)
        %fwrite(handles.SerPIC,num2str(c_cuad(n,:)));
        % 01 --- cuadrado
        %fwrite(handles.SerPIC,num2str(3));
        fwrite(handles.SerPIC,'c');
```

```
        pause(1)
    end
end

if isempty(c_romb) % Verificar si hay datos para enviar
else
    for n=1:size(c_romb,1)
        %fwrite(handles.SerPIC,num2str(c_romb(n,:)));
        % 10 --- rectangulo
        %fwrite(handles.SerPIC,num2str(4));
        fwrite(handles.SerPIC,'1');
        pause(1)
    end
end
```

% ---------- FUNCIÓN PARA ABRIR LA COMUNICACIÓN SERIAL----------

```
function usb_b_Callback(hObject, eventdata, handles)
set(hObject,'Enable','off')
com=get(handles.com_e,'String');
if isempty(com)||isnan(com)
    return
end
%*-*-Configuración del puerto serial*-*-*-*-*-*-*-*
handles.SerPIC = serial(['COM',com]);%Seleccionar puerto COM1
set(handles.SerPIC,'BaudRate',9600);%Velocidad 9600 baudios
set(handles.SerPIC,'DataBits',8);%8 bits de datos
set(handles.SerPIC,'Parity','none');%Sin control de paridad
set(handles.SerPIC,'StopBits',1);%Un bit de parada
set(handles.SerPIC,'FlowControl','none');%Sin control de flujo
% Abrir el puerto serial
fopen(handles.SerPIC);
% set(hObject,'Enable','on')
msgbox(['Puerto COM',com,' ABIERTO'],'USB-CDC')
% set(handles.inicio_b,'Enable','on');
% set(handles.parar_b,'Enable','off');
guidata(hObject,handles)
```

% --- RUTINA QUE SE EJECUTA AL CERRAR LA GUI.

```
function figure1_DeleteFcn(hObject, eventdata, handles)
try
    fclose(handles.SerPIC);
catch
end
delete(hObject);
```

Apéndice B

Plan de Negocios

Resumen Ejecutivo

El desarrollo tecnológico llamado “Selección y manipulación de objeto a partir de su forma geométrica y color por medio de un brazo robótico KUKA-KR16” es un proyecto de reciente creación que tiene como objetivo manipular ciertos objetos seleccionados para su clasificacion por medio de una cámara y un programa para la detección de forma regular o irregular, el cual tiene una gran proyección en la región así como en todo el país a causa de la creciente necesidad de automatizaciones redituables en lo que a corto, mediano y largo plazo se refiere, además sin dejar de lado su costo-beneficio el cual tiene un impacto inmediato en las areas que se le quiera utilizar, ya sea industrial (Regionalmente en la industria del calzado), educativa (Escuelas de todos los niveles), hogar (Domotica), o en al área de servicios debido a que todos sus componentes excepto el brazo manipulador y gripper fueron diseñados e implementados por su propio creador; su continuidad y desarrollo se basará en los conocimientos y experiencia adquiridos de su creador tanto en los demás proyectos realizados previamente, así como en el actual, correspondientes a las areas de óptica, robótica, control, computación, electrónica, etc.

Inicialmente, este proyecto se limitara de inicio a la manipulación de objetos con un peso no mayor a 16 Kg y a la sujeción de objetos con un solo tipo de gripper, el cual podría cambiarse para ampliar la gama de aplicaciones.

MISION

Automatizar todo tipo de procesos realizados por el ser humano que resulten difíciles cansadas.

VISION

Llegar a ser una empresa líder en todo el país para el área de la automatización basada en la robótica, y llegar a tener un nicho de mercado en areas como la industria, educación, hogar y servicios.

Análisis FODA

Fortalezas

Ubicación geográfica

Para garantizar una atención rápida y efectiva ante posibles fallas en las automatizaciones realizadas, se considera que debido a que los alcances iniciales de la empresa son regionales, se toma en cuenta como parámetro máximo ciudades que se encuentren a lo mucho a 3 horas en coche de León, GTO.

Iniciativa

Una de las habilidades esenciales de esta empresa con filosofía emprendedora es la iniciativa debido a que es la herramienta fundamental si se quiere llegar algo realmente innovador.

Empatía

Para lograr comprender las necesidades reales del cliente es necesario saber en qué situación se encuentra en el momento en son requeridos nuestros servicios.

Toma de decisiones

Empresa con capacidad de toma de decisiones bajo presión, lo que puede llevar a elegir la mejor alternativa de solución a cada problema bajo cualquier tipo de circunstancias difíciles.

Planeación

Para lograr un buen resultado en cuanto a efectividad de entrega (optimizando tiempos de trabajo) y dejar satisfecho al cliente para poder lograr futuras recomendaciones en los diversos sectores que se tiene por objetivo, es necesaria una buena planeación.

Pensamiento crítico

En el medio de los desarrollos tecnológicos se está expuesto a demasiada información que debe ser evaluada y criticada para poder discernir entre la información útil y la que no lo es. La habilidad de tener un pensamiento crítico nos permitirá poder realizar dicha evaluación en una forma eficiente para poder lograr el objetivo principal de vender un buen producto y que el cliente quede satisfecho.

Investigación

Nunca se debe dejar de aprender, debido a esto la habilidad para poder realizar investigaciones que permitan conocer aquello que le pueda faltar para mejorar la automatización es algo esencial. Una inteligente y oportuna investigación podrá conducir muchas veces al éxito.

Ventajas competitivas

1. Precio accesible debido a que los componentes son fabricados por su creador.
2. Inversión inicial con un capital mínimo (computadora personal, software gratuito, equipo de desarrollo para pequeños prototipos de prueba).
3. Tipo de empresa el cual tiene como premisa la innovación tecnológica por medio de los robots, lo que la hace única en la región.

Debilidades

Experiencia

Empresa con ideas innovadores pero con poca experiencia en el medio.

Robustez

Capacidad de atención inicial reducida debido a que su creador será el único trabajador y realizara diversas tareas.

Reputación

Empresa nueva que tendrá que irse ganado la reputación con la buena atención y calidad de los trabajos.

Oportunidades

Innovación

Productos de desarrollo tecnológico para la realización de tareas tediosas y cansadas para el ser humano nunca antes realizados, debido a esto, es una gran oportunidad para abarcar un mercado que no cuenta con muchos competidores.

Tendencias de industrias y estilo de vida

Debido a que los procesos realizados en industrias, escuelas y trabajos de la mayoría de las personas consumen casi gran parte del tiempo disponible y en veces hasta el libre, esta variable toma un papel trascendental, por lo que este tipo de empresa crea proyectos que ayuden a facilitar dichas tareas y optimizar tiempos, logrando una mayor eficiencia.

Mercado objetivo

La región en la que se planea iniciar esta empresa es una zona industrial de las más importantes del país en el rubro del cuero-calzado por lo que es un nicho en el cual se puede aplicar la robótica como medio para la automatización de procesos.

Amenazas

Demanda del mercado

La demanda es un factor a considerar debido a que inicialmente se espera muy poca debido que es una empresa poco conocida, teniendo como objetivo que dicha demanda valla aumentando con los buenos productos y precios que se ofrecen.

Respaldo financiero

El respaldo financiero inicial es muy poco, por lo que una amenaza para el desarrollo de esta empresa es que no se genere dicho respaldo financiero a corto plazo.

FINANZAS

DISPOSITIVO	DESCRIPCION	CANTIDAD	COSTO
SISTEMA ROBOTICO KUKA KR16	Manipulador programable	1	$ 900,000.0
Interfaz de Comunicación USB CDC	Elemento necesario para establecer comunicación de una PC a cualquier dispositivo con entradas Digitales.	1	$ 1500.00
Software (manual) de reconocimiento de forma	Algoritmo semi-automatico en procesamiento de imagen y comunicación, utilizado para reconocer formas geometricas	1	$25,000.00
Software (automático) en tiempo real de reconocimiento de forma	Algoritmo totalmente automático en los pasos para reconocimiento y comunicación con elementos que cuenten con entradas digitales.	1	$40,000.00
Software del manipulador	Algoritmo capaz de controlar trayectorias del manipulador a partir de la recepción de señales externas.	1	$30,000.00

Mano de obra		x	$40,000.00
Instalación	Calibración del sistema	x	$25,000.00
TOTAL		1061,500.00	

Cabe señalar que se puede usar un brazo manipulador a elección del comprador y no necesariamente el propuesto, por lo que los costos pudieran variar en función a ese elemento.